Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Continuous Assessment for Laboratory / Assignment sessions

Academic Year 2022-23

Name: Kushal Patel SAP ID: 60004210058

Course: Computer Networks Course Code: **DJ19CEL405**

Year: **S.Y. B.Tech.** Sem: **IV** Batch: A2

Department: Computer Engineering

Performance Indicators (Any no. of Indicators) (Maximum 5 marks per indicator)	1	2	3	4	5	6	7	8	9	10	11	Σ	Avg	A1	A2	Σ	Avg
Course Outcome	**1**	**1**	**2**	**2**	**4**	**3**	**4**	**4**	**5**	**4**	**6**						
1. Knowledge (Factual/Conceptual/Procedural/ Metacognitive)	03	3	3	3	3	3			3	3	3			1	1		
2. Describe (Factual/Conceptual/Procedural/ Metacognitive)	03	3	3	3	3	3			3	3	3			1	1		
3. Demonstration (Factual/Conceptual/Procedural/ Metacognitive)	04	4	4	5	4	4			4	4	4			1	1		
4. Strategy (Analyse & / or Evaluate) (Factual/Conceptual/ Procedural/Metacognitive)	04	4	5	5	4	4			4	4	4			1	1		
5. Interpret/ Develop (Factual/Conceptual/ Procedural/Metacognitive)	–	–	–	–	–	–	–	–	–					–	–		
6. Attitude towards learning (receiving, attending, responding, valuing, organizing, characterization by value)	04	5	5	5	3	4			4	3	4			1	1		
7. Non-verbal communication skills/ Behaviour or Behavioural skills (motor skills, hand-eye coordination, gross body movements, finely coordinated body movements speech behaviours)	–	–	–	–	–	–	–	–	–					–	–		
Total	18	19	20	21	17	18			18	17	18			5	5		
Signature of the faculty member																	

Outstanding (5), Excellent (4), Good (3), Fair (2), Needs Improvement (1)

Laboratory marks Σ Avg. =	Assignment marks Σ Avg. =	Total Term-work (25) =
Laboratory Scaled to (15) =	Assignment Scaled to (10) =	Sign of the Student:

Signature of the Faculty member:
Name of the Faculty member:

Signature of Head of the Department
Date:

Department of Computer Engineering

Class: S.Y. B.Tech. **Semester: IV**

Course Code: DJ19CEL405 **Course Name: Computer Networks Lab**

Name: Kushal Patel

SAP ID: 60004210058

Experiment No: 1

Aim: To study different networking devices and topologies.

Theory: Network devices, also known as networking hardware, are physical devices that allow hardware on a computer network to communicate and interact with one another. For example Repeater, Hub, Bridge, Switch, Routers, Gateway, Brouter etc.

Networking Devices:

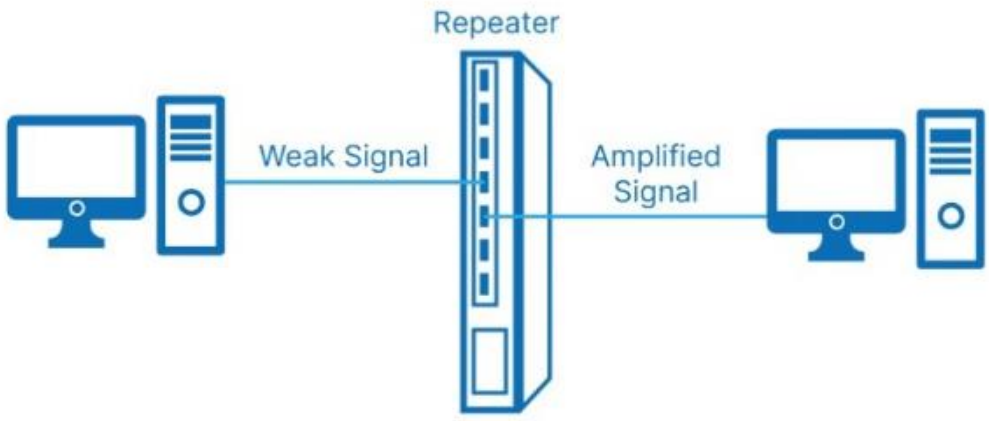
1. Repeater:

i. Introduction:

In computer networking, a repeater is a device that regenerates a network signal that has become weak or distorted due to distance or interference. It receives a signal, amplifies it, and then transmits the signal to the next device in the network. Repeaters are used to extend the range of a network by boosting the signal, allowing data to travel further without being lost or corrupted.

Repeaters are commonly used in Ethernet and other types of wired networks, as well as wireless networks. In wired networks, repeaters are often used to extend the reach of a network beyond the standard maximum cable length.

ii. Diagram:



iii. Advantages:

- The Repeater supports the signal strength.
- These Repeaters are both cheap and easy to use.
- The Repeaters have no impact on the network's performance.
- These Repeaters are capable of retransmitting data and boosting weak signals.

iv. Disadvantages:

- The number of collisions increases as the number of repeaters increases.
- Only a limited number of repeaters can be connected to it.
- We can't connect the different network architectures in repeaters.
- The data traffic in the Repeaters cannot be reduced.

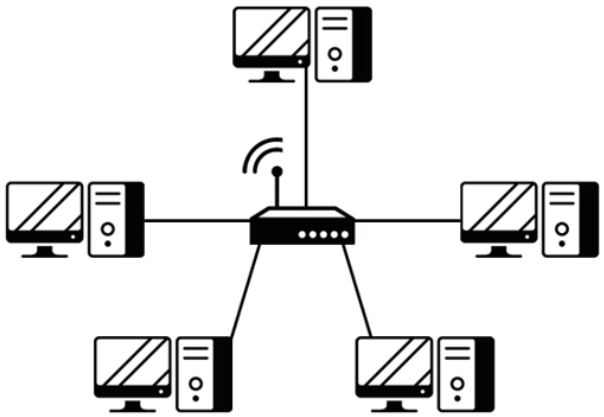
2.Hub

i. Introduction:

A hub is a physical layer networking device which is used to connect multiple devices in a network. They are generally used to connect computers in a LAN.

A hub has many ports in it. A computer which intends to be connected to the network is plugged in to one of these ports. When a data frame arrives at a port, it is broadcast to every other port, without considering whether it is destined for a particular destination or not.

ii. Diagram



iii. Advantages

- Easy to install
- Very little delay
- It is used for internal connectivity between the system
- Can different media type
- Cheaper
- Hub device does not affect the performance of the network seriously
- It can extend the total distance of the network

iv. Disadvantages

- It runs half-duplex
- Can not filter information
- It can not connect different type of network architecture such as a token ring and Ethernet extra
- It does not have a mechanism to reduce the network traffic
- Passes packet to all connected segment
- Can not reduce network traffic
- It will broadcast to all the port
- Extend the collision

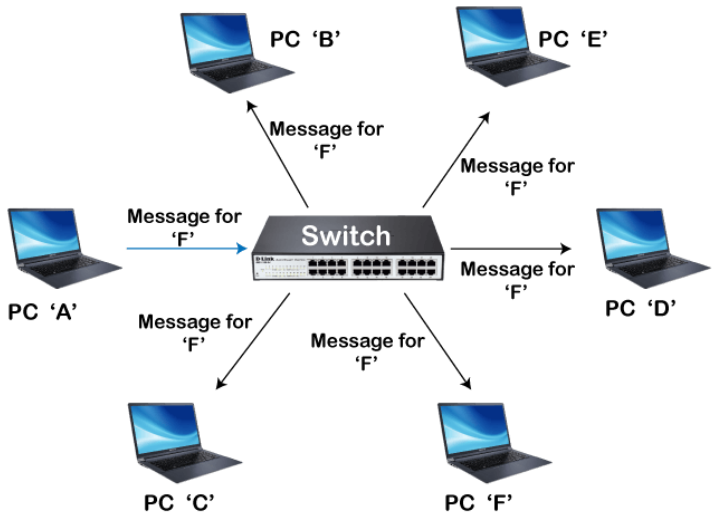
3. Switch

i. Introduction

A switch is a data link layer networking device which connects devices in a network and uses packet switching to send and receive data over the network.

Like a hub, a switch also has many ports, to which computers are plugged in. However, when a data frame arrives at any port of a network switch, it examines the destination address and sends the frame to the corresponding device(s). Thus, it supports both unicast and multicast communications.

ii. Diagram



iii. Advantages

- They increase the available bandwidth of the network.
- They help in reducing workload on individual host PCs
- They increase the performance of the network.
- Networks which use switches will have less frame collisions. This is due to the fact that switches create collision domains for each connection.
- Switches can be connected directly to workstations.

iv. Disadvantages

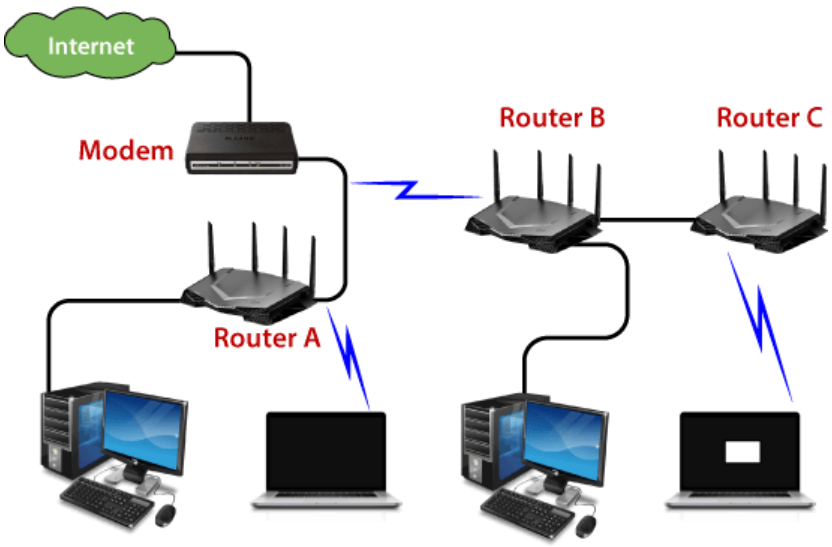
- They are more expensive compare to network bridges.
- Network connectivity issues are difficult to be traced through the network switch.
- Broadcast traffic may be troublesome.
- If switches are in promiscuous mode, they are vulnerable to security attacks e.g. spoofing IP address or capturing of ethernet frames.
- Proper design and configuration is needed in order to handle multicast packets.

4. Router

i. Introduction

A router is a device that connects two or more packet-switched networks or subnetworks. It serves two primary functions: managing traffic between these networks by forwarding data packets to their intended IP addresses, and allowing multiple devices to use the same Internet connection.

ii. Diagram



iii. Advantages

- It provides sophisticated routing, flow control, and traffic isolation
- Reduce network traffic by creating collision domains
- Reduce network traffic by creating broadcast domains
- Can connect different network architecture, such as Ethernet and token ring
- It can choose the best path across the internetwork using dynamic routing algorithms
- Allows achieving loops so that redundant paths are available

iv. Disadvantages

- A router is more expensive than bridge or repeaters
- Router only work with rotatable network protocol, not all protocol are routable
- Dynamic router communication causes additional network traffic
- Are relatively complex device
- Can require a considerable amount of initial configuration

5. Modem

i. Introduction

A modem, also called a modulator-demodulator, modulates digital signals to analog signals transmitted over telephone lines and also demodulates such analog signals to digital signals.

ii. Diagram

iii. Advantages

- More useful in connecting LAN with the internet
- Speed depends on the cost
- Slow speed when compared to the hub
- A limited number of a system can be connected
- A modem is most probably widely used in data communication roadway
- A modem converts that the digital signal into an analog signal

iv. Disadvantages

- Acts just as an interface between LAN and internet
- No traffic maintenance is present
- A modem is not understood the intermediate process
- The modem does not know about the own destination path

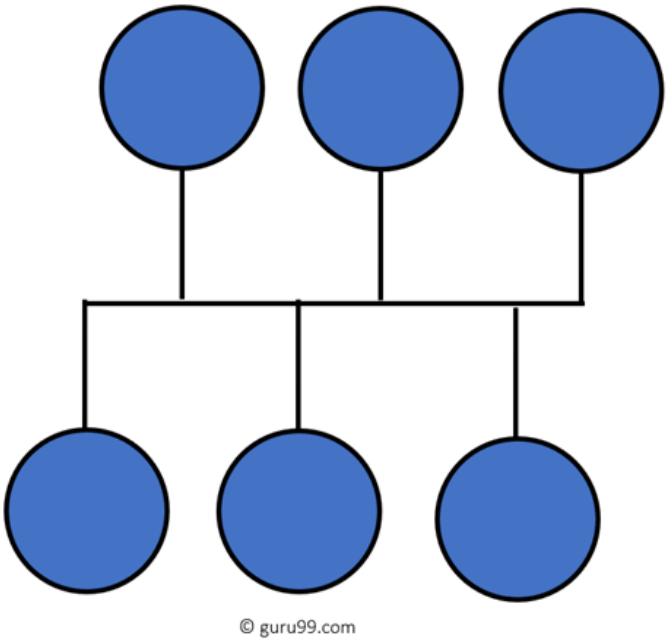
Networking Topologies:

1. Bus:

i. Introduction:

Bus topology uses a single cable which connects all the included nodes. The main cable acts as a spine for the entire network. One of the computers in the network acts as the computer server. When it has two endpoints, it is known as a linear bus topology

ii. Diagram:



iii. Advantages:

- Cost of the cable is very less as compared to other topology, so it is widely used to build small networks.
- Famous for LAN network because they are inexpensive and easy to install.
- It is widely used when a network installation is small, simple, or temporary.
- It is one of the passive topologies. So computers on the bus only listen for data being sent, that are not responsible for moving the data from one computer to others

iv. Disadvantages:

- In case if the common cable fails, then the entire system will crash down.
- When network traffic is heavy, it develops collisions in the network.
- Whenever network traffic is heavy, or nodes are too many, the performance time of the network significantly decreases.
- Cables are always of a limited length.

V. Applications:

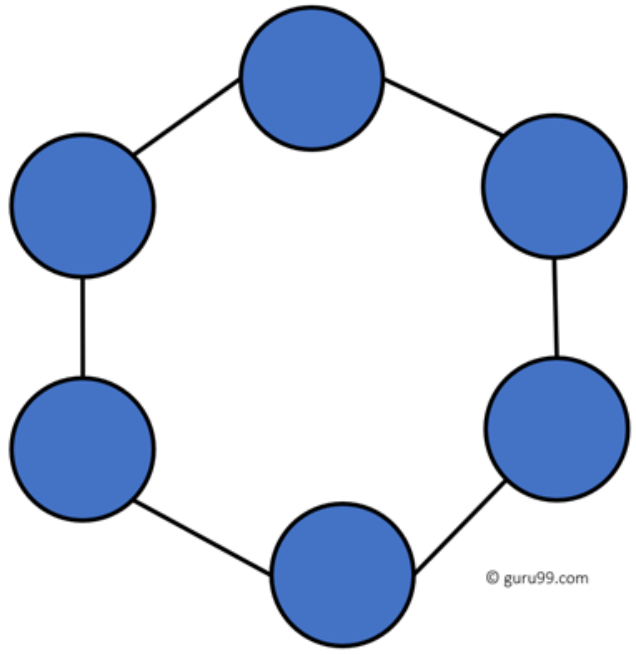
- A bus topology is used to connect two floors using a single line.
- A bus topology is used by an Ethernet network
- In this type of network topology, one computer works like a server whereas the other works as a client.
- The main function of the server is to exchange information between different client computers.
- Bus topology network is used to add the printers, I/O devices in the offices or home

2. Ring

i. Introduction

In a ring network, every device has exactly two neighboring devices for communication purpose. It is called a ring topology as its formation is like a ring. In this topology, every computer is connected to another computer. Here, the last node is combined with a first one. This topology uses token to pass the information from one computer to another. In this topology, all the messages travel through a ring in the same direction.

ii. Diagram



iii. Advantages

- Easy to install and reconfigure.
- Adding or deleting a device in-ring topology needs you to move only two connections.
- The troubleshooting process is difficult in a ring topology.

- Failure of one computer can disturb the whole network.
- Offers equal access to all the computers of the networks
- Faster error checking and acknowledgment.

iv. Disadvantages

- Unidirectional traffic.
- Break in a single ring can risk the breaking of the entire network
- Modern days high-speed LANs made this topology less popular.
- In the ring, topology signals are circulating at all times, which develops unwanted power consumption.
- It is very difficult to troubleshoot the ring network.
- Adding or removing the computers can disturb the network activity.

V. Applications

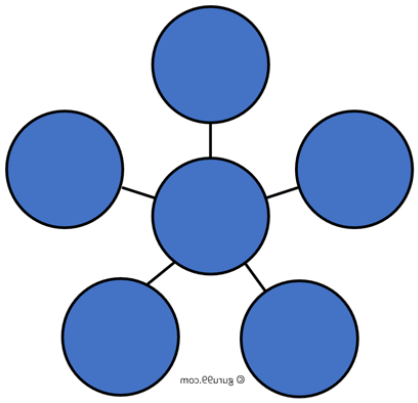
Ring network topologies are used when a simple network is needed. They are suitable for locations that do not rely on very high data-transfer speeds, and where the network is unlikely to alter in size or structure. For example, a small office with only a few nodes may use a ring network topology.

3. Star

i. Introduction

In the star topology, all the computers connect with the help of a hub. This cable is called a central node, and all other nodes are connected using this central node. It is most popular on LAN networks as they are inexpensive and easy to install.

ii. Diagram



iii. Advantages

- Easy to troubleshoot, set up, and modify.
- Only those nodes are affected, that has failed. Other nodes still work.
- Fast performance with few nodes and very low network traffic.
- In Star topology, addition, deletion, and moving of the devices are easy.

iv. Disadvantages

- If the hub or concentrator fails, attached nodes are disabled.
- Cost of installation of star topology is costly.
- Heavy network traffic can sometimes slow the bus considerably.
- Performance depends on the hub's capacity
- A damaged cable or lack of proper termination may bring the network down.

v. Applications

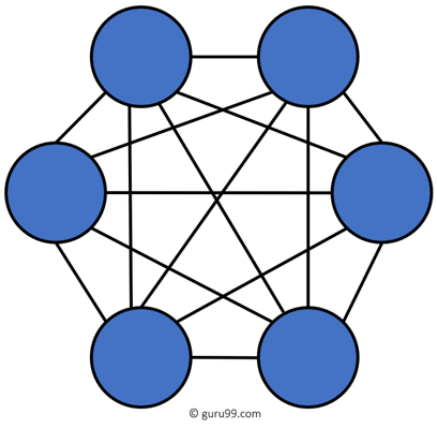
Nowadays, many institutes, airports, hospitals, and banks are places where you can easily find star topology as the most commonly used network connection. In star topology, each device in the network completely depends on the central hub, i.e. if the hub fails, the whole network fails.

4. Mesh

i. Introduction

The mesh topology has a unique network design in which each computer on the network connects to every other. It is develops a P2P (point-to-point) connection between all the devices of the network. It offers a high level of redundancy, so even if one network cable fails, still data has an alternative path to reach its destination.

ii. Diagram



iii. Advantages

- The network can be expanded without disrupting current users.
- Need extra capable compared with other LAN topologies.
- No traffic problem as nodes has dedicated links.
- Dedicated links help you to eliminate the traffic problem.
- A mesh topology is robust.
- It has multiple links, so if any single route is blocked, then other routes should be used for data communication.
- P2P links make the fault identification isolation process easy.
- It helps you to avoid the chances of network failure by connecting all the systems to a central node.
- Every system has its privacy and security.

iv. Disadvantages

- Installation is complex because every node is connected to every node.
- It is expensive due to the use of more cables. No proper utilization of systems.
- Complicated implementation.
- It requires more space for dedicated links.
- Because of the amount of cabling and the number of input-outputs, it is expensive to implement.
- It requires a large space to run the cables.

v. Applications

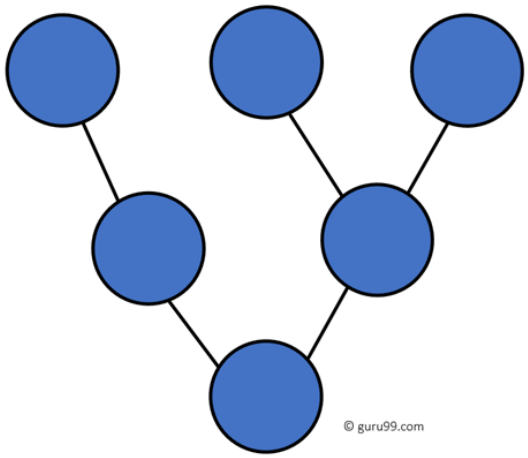
- military organisations often use mesh topologies to avoid breakdowns in communication
- cities are increasingly using wireless mesh networks to help monitor traffic flow, sewage treatment and to help control street lighting
- emergency services, such as police and fire services, also use wireless mesh networks to ensure that communication is reliable
- some utility companies who provide gas and electric use mesh networks to allow smart meters to send readings automatically.

5. Tree

i. Introduction

Tree topologies have a root node, and all other nodes are connected which form a hierarchy. So it is also known as hierarchical topology. This topology integrates various star topologies together in a single bus, so it is known as a Star Bus topology. Tree topology is a very common network which is similar to a bus and star topology.

ii. Diagram



iii. Advantages

- Failure of one node never affects the rest of the network.
- Node expansion is fast and easy.
- Detection of error is an easy process
- It is easy to manage and maintain

iv. Disadvantages

- It is heavily cabled topology
- If more nodes are added, then its maintenance is difficult
- If the hub or concentrator fails, attached nodes are also disabled.

v. Applications

Tree topology is often used to connect multiple devices, such as printers and computers, in a home or small office. Tree topology is commonly used to create bus networks, where each device is connected to a central server.

Conclusion:

A computer network is a group of two or more interconnected computer systems.Computer networks help you to connect with multiple computers together to send and receive information.Switches work as a controller which connects computers, printers, and other hardware devices.Routers help you to connect with multiple networks. It enables you to share a single internet connection and saves money.Servers are computers that hold shared programs, files, and the network operating system.Clients are computer device which accesses and uses the network and shares network resources.Hub is a device that split a network connection into multiple computers.Access points allow devices to connect to the wireless network without cables.Network Interface card sends, receives data and controls data flow between the computer and the network.A protocol is the set of defined rules which that allows two entities to communicate across the network.

Thus, we can decide on choosing the network devices on the basis of a few factors like size of the network, scalability, cost, management and security requirements. For example, for a small network, a single router may be sufficient, but for a larger one, you may need multiple routers, switches, access points, etc. Similarly, If your network is likely to grow in the future, you should consider devices that can be easily expanded, such as modular switches or routers. Finally, you should choose your Networking device on the basis of your needs.

Department of Computer Engineering

Class: S.Y. B.Tech. **Semester: IV**

Course Code: DJ19CEL405 **Course Name: Computer Networks Lab**

Name: Kushal Patel **SAP ID:60004210058**

Experiment No: 2

Aim: To study different networking commands.

Theory:

1. **Ifconfig/ipconfig:**
   i. **Introduction:**
   Ipconfig stands for Ïnternet Protocol Configuration is a console application program of some computer operating systems that displays all current TCP/IP network configuration values and refreshes Dynamic Configurations Protocols(DHCP) and Domain Name System(DNS) settings.
   ii. Output:

```
Microsoft Windows [Version 10.0.19044.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Users\djsce.student>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::828:2fa9:11ec:1401%10
   IPv4 Address. . . . . . . . . . . : 10.120.63.84
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.120.63.1

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter VMware Network Adapter VMnet1:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::85d6:2798:7077:b8f2%6
   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::85d6:2798:7077:b8f2%6
   IPv4 Address. . . . . . . . . . . : 192.168.41.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet8:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::1186:aaa5:de15:4534%9
   IPv4 Address. . . . . . . . . . . : 192.168.5.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::aceb:7bec:8458:a3af%7
   IPv4 Address. . . . . . . . . . . : 10.120.113.148
   Subnet Mask . . . . . . . . . . . : 255.255.254.0
   Default Gateway . . . . . . . . . : 10.120.112.1

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

C:\Users\djsce.student>
```

Conclusion: Hence ipconfig is studied and executed successfully.

2. **Netstat:**
   - i. **Introduction:** When using this tool, you can list active networks (incoming and outgoing) connections and listening ports. You can view network adapter statistics as well as statistics for protocols (such as IPv4 and IPv6). You can even display the current routing table, and much more.
   - ii. **Output:**

```
C:\Users\djsce.student>netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    10.120.113.148:58218   w-srv-sccm:10123       ESTABLISHED
  TCP    10.120.113.148:58396   20.198.118.190:https   ESTABLISHED
  TCP    10.120.113.148:58398   se-in-f188:5228        ESTABLISHED
  TCP    10.120.113.148:58761   52.114.40.57:https     ESTABLISHED
  TCP    10.120.113.148:58769   52.114.40.57:https     ESTABLISHED
  TCP    10.120.113.148:58882   a23-35-6-201:https     CLOSE_WAIT
  TCP    10.120.113.148:58995   13.107.6.171:https     ESTABLISHED
  TCP    10.120.113.148:59008   13.107.21.200:https    CLOSE_WAIT
  TCP    10.120.113.148:59011   20.111.38.59:https     CLOSE_WAIT
  TCP    10.120.113.148:59014   13.107.3.254:https     CLOSE_WAIT
  TCP    10.120.113.148:59015   204.79.197.222:https   CLOSE_WAIT
  TCP    10.120.113.148:59075   a23-35-6-201:https     ESTABLISHED
  TCP    10.120.113.148:59077   20.50.201.200:https    ESTABLISHED
  TCP    10.120.113.148:59086   199.232.22.132:https   ESTABLISHED
  TCP    10.120.113.148:59109   52.178.17.3:https      ESTABLISHED
  TCP    10.120.113.148:59114   ec2-3-6-18-84:https    TIME_WAIT
  TCP    10.120.113.148:59118   172.64.137.23:https    ESTABLISHED
  TCP    10.120.113.148:59121   172.64.136.23:https    ESTABLISHED
  TCP    10.120.113.148:59123   bom12s12-in-f14:https  ESTABLISHED
  TCP    10.120.113.148:59124   bom07s45-in-f10:https  ESTABLISHED
```

   - iii. **Conclusion: Hence netstat. is studied and executed successfully.**

3.Ping:

i. **Introduction:** The ping command is a Command Prompt command used to test the ability of the source computer to reach a specified destination computer. It's a simple way to verify that a computer can communicate with another computer or network device.

The ping command operates by sending Internet Control Message Protocol (ICMP) Echo Request messages to the destination computer and waiting for a response. The two major pieces of information that the ping command provides are how many of those responses are returned and how long it takes for them to return.

For example, you might find no responses when pinging a network printer, only to find out that the printer is offline and its cable needs replaced. Or maybe you need to ping a router to verify that your computer can connect to it to eliminate it as a possible cause for a networking issue.

iii. **OUTPUT:**

```
C:\Users\djsce.student>ping

Usage: ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS]
            [-r count] [-s count] [[-j host-list] | [-k host-list]]
            [-w timeout] [-R] [-S srcaddr] [-c compartment] [-p]
            [-4] [-6] target_name

Options:
    -t             Ping the specified host until stopped.
                   To see statistics and continue - type Control-Break;
                   To stop - type Control-C.
    -a             Resolve addresses to hostnames.
    -n count       Number of echo requests to send.
    -l size        Send buffer size.
    -f             Set Don't Fragment flag in packet (IPv4-only).
    -i TTL         Time To Live.
    -v TOS         Type Of Service (IPv4-only. This setting has been deprecated
                   and has no effect on the type of service field in the IP
                   Header).
    -r count       Record route for count hops (IPv4-only).
    -s count       Timestamp for count hops (IPv4-only).
    -j host-list   Loose source route along host-list (IPv4-only).
    -k host-list   Strict source route along host-list (IPv4-only).
    -w timeout     Timeout in milliseconds to wait for each reply.
    -R             Use routing header to test reverse route also (IPv6-only).
                   Per RFC 5095 the use of this routing header has been
                   deprecated. Some systems may drop echo requests if
                   this header is used.
    -S srcaddr     Source address to use.
    -c compartment Routing compartment identifier.
    -p             Ping a Hyper-V Network Virtualization provider address.
    -4             Force using IPv4.
    -6             Force using IPv6.
```

Conclusion: Hence ping is studied and executed successfully.

4. pathping:

i. Introduction: The command provides details of the path between two hosts and ping-like statistics for each node in the path based on samples taken over a time period, depending on how many nodes are between the start and end host.

The advantages of pathping over ping and tracepath are that each node is pinged as the result of a single command, and that the behavior of nodes is studied over an extended time period, rather than the default ping sample of four messages or default traceroute single route trace. The disadvantage is that it takes a total of 25 seconds per hop to show the ping statistics.[4]

ii. **Output:**

```
C:\Users\djsce.student>pathping

Usage: pathping [-g host-list] [-h maximum_hops] [-i address] [-n]
                [-p period] [-q num_queries] [-w timeout]
                [-4] [-6] target_name

Options:
    -g host-list     Loose source route along host-list.
    -h maximum_hops  Maximum number of hops to search for target.
    -i address       Use the specified source address.
    -n               Do not resolve addresses to hostnames.
    -p period        Wait period milliseconds between pings.
    -q num_queries   Number of queries per hop.
    -w timeout       Wait timeout milliseconds for each reply.
    -4               Force using IPv4.
    -6               Force using IPv6.

C:\Users\djsce.student>
```

iii. Conclusion: Hence pathping is studied and executed successfully.

5: nslookup:

i. introductions: Microsoft Windows includes a tool called NSLOOKUP that you can use via the command prompt. This tool can be used to check DNS records propagation and resolution using different servers, and perform other troubleshooting steps.

Displays information that you can use to diagnose Domain Name System (DNS) infrastructure. Before using this tool, you should be familiar with how DNS works. The nslookup command-line tool is available only if you have installed the TCP/IP protocol.

The nslookup command-line tool has two modes: interactive and noninteractive.

If you need to look up only a single piece of data, we recommend using the non-interactive mode. For the first parameter, type the name or IP address of the computer that you want to look up. For the second parameter, type the name or IP address of a DNS name server. If you omit the second argument, **nslookup** uses the default DNS name server.

If you need to look up more than one piece of data, you can use interactive mode. Type a hyphen (-) for the first parameter and the name or IP address of a DNS name server for the second parameter. If you omit both parameters, the tool uses the default DNS name server. While using the interactive mode, you can:

ii. Output:

```
C:\Users\djsce.student>nslookup
Default Server:  MUMDC-PRIM.SVKMGRP.COM
Address:  192.168.2.51
```

Conclusion: Hence nslookup is studied and executed successfully.

6. hostname:

i. introduction: Displays the host name portion of the full computer name of the computer.

Sometimes, you require the hostname of your computer system when it connects with a network. The hostname helps the other devices to find your computer on that network. You can easily get the hostname of your computer system by typing a simple command on command prompt (CMD) in the Windows Operating system

ii. output:

```
C:\Users\djsce.student>hostname
MUM0922CPU0391
```

Conclusion: Hence hostname is studied and executed successfully.

7. tracert (-d,-h,-w):

i. introduction: This diagnostic tool determines the path taken to a destination by sending Internet Control Message Protocol (ICMP) echo Request or ICMPv6 messages to the destination with incrementally increasing time to live (TTL) field values. Each router along the path is required to decrement the TTL in an IP packet by at least 1 before forwarding it. Effectively, the TTL is a maximum link counter. When the TTL on a packet reaches 0, the router is expected to return an ICMP time Exceeded message to the source computer.

This command determines the path by sending the first echo Request message with a TTL of 1 and incrementing the TTL by 1 on each subsequent transmission until the target responds or the maximum number of hops is reached. The maximum number of hops is 30 by default and can be specified using the **/h** parameter.

The path is determined by examining the ICMP time Exceeded messages returned by intermediate routers and the echo Reply message returned by the destination. However, some routers do not return time Exceeded messages for packets with expired TTL values and are invisible to the **tracert** command. In this case, a row of asterisks (*) is displayed for that hop. The path displayed is the list of near/side router interfaces of the routers in the path between a source host and a destination. The near/side interface is the interface of the router that is closest to the sending host in the path.

ii. output:

```
C:\Users\djsce.student>tracert

Usage: tracert [-d] [-h maximum_hops] [-j host-list] [-w timeout]
               [-R] [-S srcaddr] [-4] [-6] target_name

Options:
    -d                  Do not resolve addresses to hostnames.
    -h maximum_hops     Maximum number of hops to search for target.
    -j host-list        Loose source route along host-list (IPv4-only).
    -w timeout          Wait timeout milliseconds for each reply.
    -R                  Trace round-trip path (IPv6-only).
    -S srcaddr          Source address to use (IPv6-only).
    -4                  Force using IPv4.
    -6                  Force using IPv6.
```

Conclusion: Hence tracert is studied and executed successfully.

8. arp(-a,-n):

i. introduction: Displays and modifies entries in the Address Resolution Protocol (ARP) cache. The ARP cache contains one or more tables that are used to store IP addresses and their resolved Ethernet or Token Ring physical addresses. There is a separate table for each Ethernet or Token Ring network adapter installed on your computer. Used without parameters, **arp** displays help information.

ii. output:

```
C:\Users\djsce.student>arp

Displays and modifies the IP-to-Physical address translation tables used by
address resolution protocol (ARP).

ARP -s inet_addr eth_addr [if_addr]
ARP -d inet_addr [if_addr]
ARP -a [inet_addr] [-N if_addr] [-v]

  -a            Displays current ARP entries by interrogating the current
                protocol data.  If inet_addr is specified, the IP and Physical
                addresses for only the specified computer are displayed.  If
                more than one network interface uses ARP, entries for each ARP
                table are displayed.
  -g            Same as -a.
  -v            Displays current ARP entries in verbose mode.  All invalid
                entries and entries on the loop-back interface will be shown.
  inet_addr     Specifies an internet address.
  -N if_addr    Displays the ARP entries for the network interface specified
                by if_addr.
  -d            Deletes the host specified by inet_addr. inet_addr may be
                wildcarded with * to delete all hosts.
  -s            Adds the host and associates the Internet address inet_addr
                with the Physical address eth_addr.  The Physical address is
                given as 6 hexadecimal bytes separated by hyphens. The entry
                is permanent.
  eth_addr      Specifies a physical address.
  if_addr       If present, this specifies the Internet address of the
                interface whose address translation table should be modified.
                If not present, the first applicable interface will be used.
Example:
  > arp -s 157.55.85.212   00-aa-00-62-c6-09  .... Adds a static entry.
  > arp -a                                    .... Displays the arp table.
```

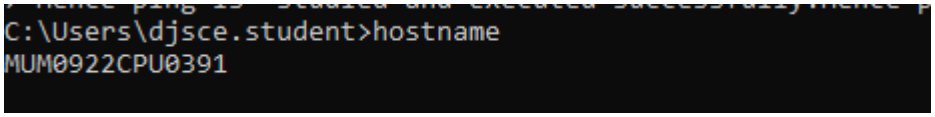
Conclusion: Hence arp is studied and executed successfully.

9. getmac:

i. introduction: Displays the host name portion of the full computer name of the computer.

Sometimes, you require the hostname of your computer system when it connects with a network. The hostname helps the other devices to find your computer on that network. You can easily get the hostname of your computer system by typing a simple command on command prompt (CMD) in the Windows Operating system

ii. output:



Conclusion: Hence getmac is studied and executed successfully.

SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)
Department of Computer Engineering

Class: S.Y. B.Tech. **Semester: IV**
Course Code: DJ19CEL405 **Course Name: Computer Networks Lab**

Name: Kushal Patel **SAP ID:60004210058**
Date of Performance: 02-03-2023 **Date of Submission: 12-03-2023**

Experiment No: 3

Aim: Write a program to implement Framing Techniques:
Character count, Byte stuffing, Bit stuffing

Your program will read a file (in pdf). Convert given files into ASCII and then Binary values whichever applicable. Use this prepared file and create frames of manageable units called frames with following three options

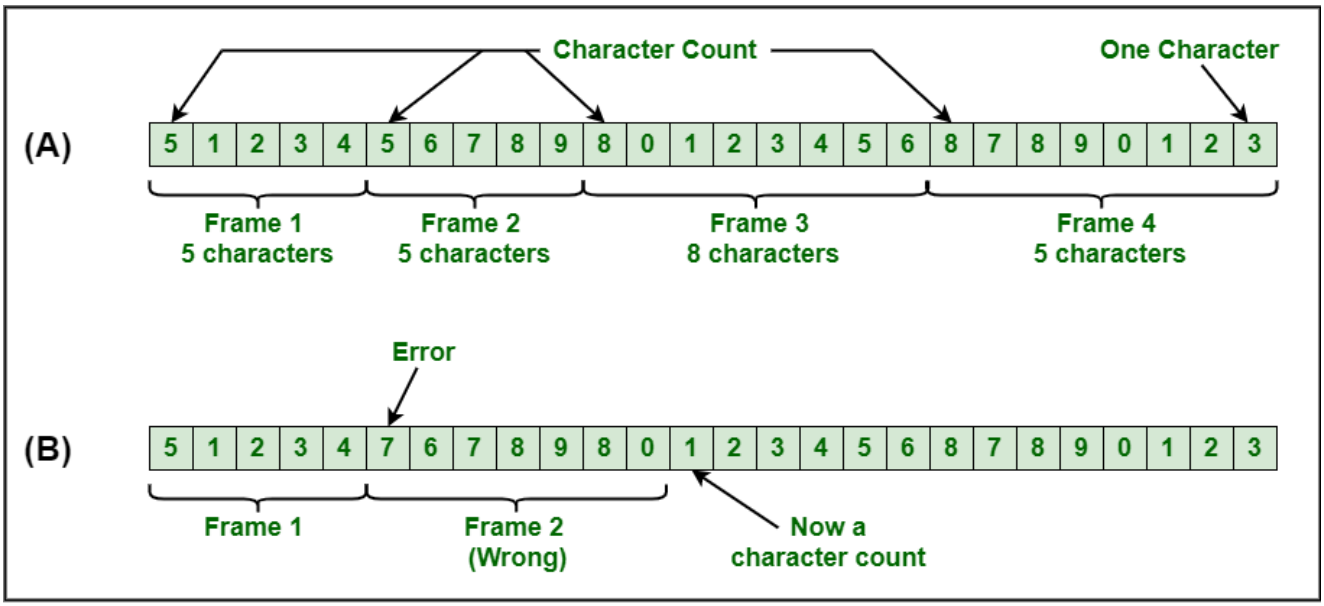
1. Character count: Randomly generate character count between 2^0 to 2^16. Display the various frames created with the help of preprocessed file as above.
2. Bit Stuffing: Generate Fixed frames of equal length as per used definded input probably Power of 2(2^n) from above preprocessed file by using bit stuffing method. Assume (01111110) as Start and End of Frame. Bit "Zero" should be stuffed in payload if 5 consecutive "1" occurs in data. Pad your data sequence (1000000…..0) to make the preprocessed file multiple of 128.
3. Byte Stuffing: Implement variable length byte stuffing using character stuffing method. Insert special character("ESC") character in preprocessed file at random interval between 2^0 to 2^16. Display the various frames created with the help of preprocessed file as above.

Theory:

Explain character count with example.

This method is rarely used and is generally required to count total number of characters that are present in frame. This is be done by using field in header. Character count method ensures data link layer at the receiver or

destination about total number of characters that follow, and about where the frame ends.
There is disadvantage also of using this method i.e., if anyhow character count is disturbed or distorted by an error occurring during transmission, then destination or receiver might lose synchronization. The destination or receiver might also be not able to locate or identify beginning of next frame.



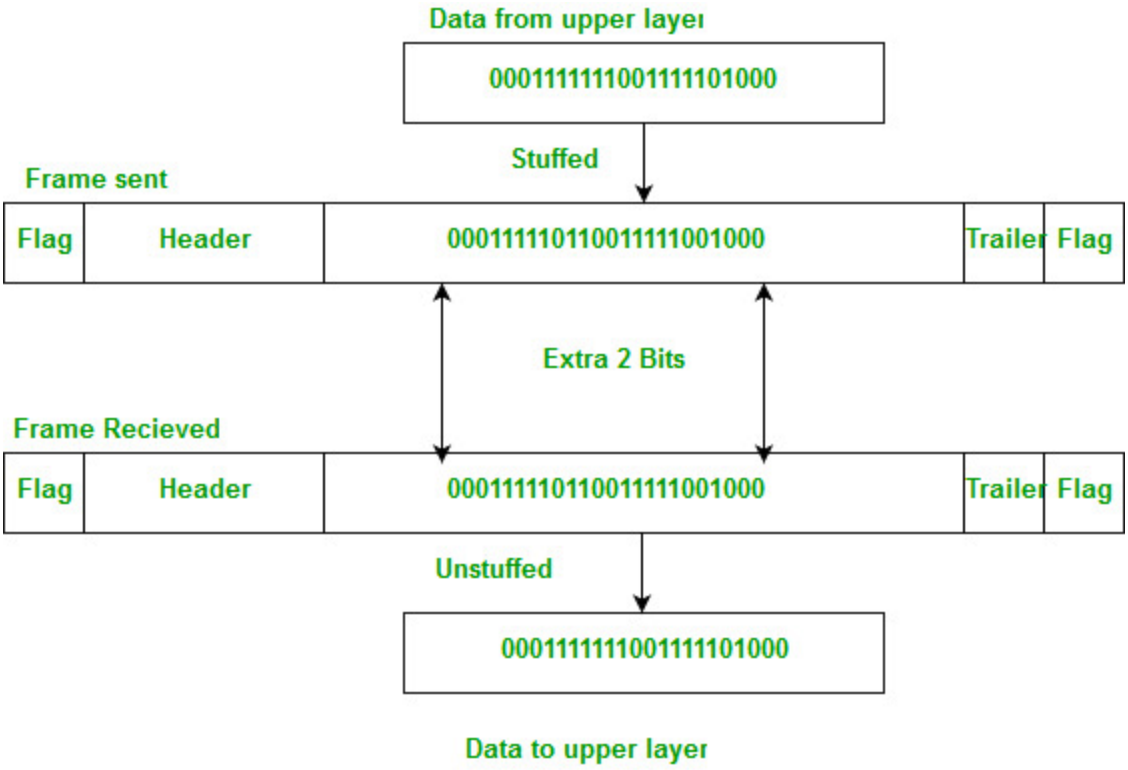
A Character Stream

(A) Without Errors
(B) With one Error

Explain Bit stuffing with example.

Bit stuffing is also known as bit-oriented framing or bit-oriented approach. In bit stuffing, extra bits are being added by network protocol designers to data streams. It is generally insertion or addition of extra bits into transmission unit or message to be transmitted as simple way to provide

and give signaling information and data to receiver and to avoid or ignore appearance of unintended or unnecessary control sequences.
It is type of protocol management simply performed to break up bit pattern that results in transmission to go out of synchronization. Bit stuffing is very essential part of transmission process in network and communication protocol. It is also required in USB.



Explain Byte stuffing with example.

- In this method, start and end of frame are recognized with the help of flag bytes. Each frames starts with and ends with a flag byte. Two consecutive flag bytes indicate the end of one frame and start of the next one. The flag bytes used in the figure 2 used is named as "ESC" flag byte.

- Aframe delimited by flag bytes. This framing method is only applicable in 8-bit character codes which are a major disadvantage of this method as not all character codes use 8-bit characters e.g. Unicode.
- Four example of byte sequences before and after stuffing:

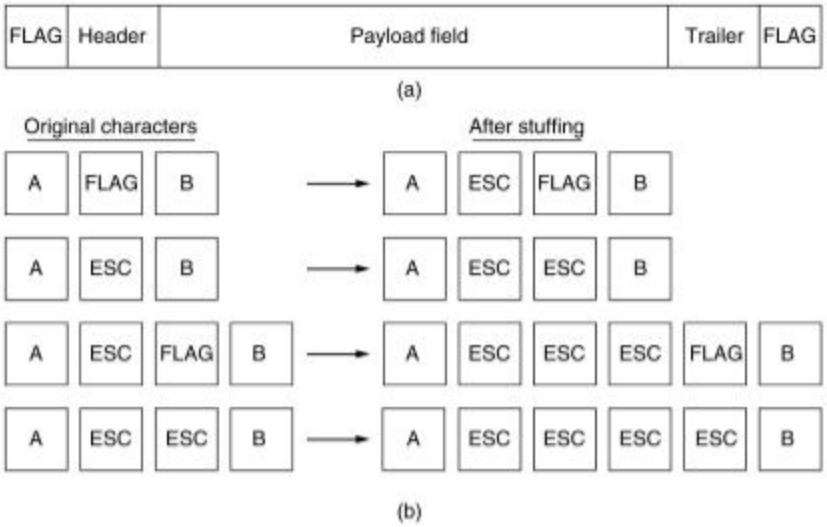


Fig2: Framing with Byte stuffing

Code:

```
import PyPDF2
import random
start = 0
end = 0
pdfFileObj = open('Lorem_ipsum.pdf', 'rb')
pdfReader = PyPDF2.PdfReader(pdfFileObj)
pageObj=pdfReader.pages[0]
file_pdf=pageObj.extract_text()
ascii_string=""

for char in file_pdf:
    ascii_string+=str(ord(char))
```

```
with open('Char_frame.txt', 'w') as f:
    c=0
    while end<len(file_pdf):
        power=random.randint(5,10)
        end += 2**power
        f.write(str(2**power)+"  "+file_pdf[start:end])
        # print(f"Frame {c}: {str(2**power)} {file_pdf[start:end]}")
        f.write("\n\n\n")
        start = end
        c += 1
        # print("total frame : ", c)
    pdfFileObj.close()


class Frame:
    data=''
    size = 0
    def __init__(self,data,n):
        self.size=n
        count=0
        framecontainer ='01111110'
        self.data += framecontainer

        for i in data:
            if i == '':
                continue
            count+=1

        if count<n:
            self.data += data
            self.data +='1'
            diff = n-count
            while diff-1 != 0:
                self.data += '0'
                diff -= 1
        else:
            self.data+=data

        self.data += framecontainer
```

```
def bitStuffing(sig):
    onecounter = 0
    index = 0
    one = []
    signal = list(sig)
    for i in signal:
        index += 1
        if i == '0':
            onecounter = 0
        else:
            onecounter += 1
        if onecounter == 5:
            one.append(index)
            onecounter = 0
    k = 0
    for i in one:
        signal.insert(i + k, '0')
        k += 1
    str1 = ""
    for ele in signal:
        str1 += ele
    return str1

binary_string=""
for char in file_pdf:
    binary_string+=(str(bin(ord(char)))[2:])
# print(binary_string)

stuffed_binary_string = bitStuffing(binary_string)

n = int(input("Enter the size of frames for bit stuffing:"))
framesize = pow(2,n)
binsize = len(stuffed_binary_string)
counted =0
framelist = []
framecount =0
```

```
while counted < binsize:
    frameinst = Frame(stuffed_binary_string[counted:framesize+counted],framesize)
    counted += framesize
    framelist.append(frameinst)
    framecount += 1

with open('Bin_frame.txt', 'w') as f:
    for inst in framelist:
        # print(inst.data,inst.size)
        f.write(str(inst.size)+"  ")
        f.write(inst.data)
        f.write("\n")
    # print(framecount, binsize)
    f.write(str(framecount)+"  "+str(binsize))
```

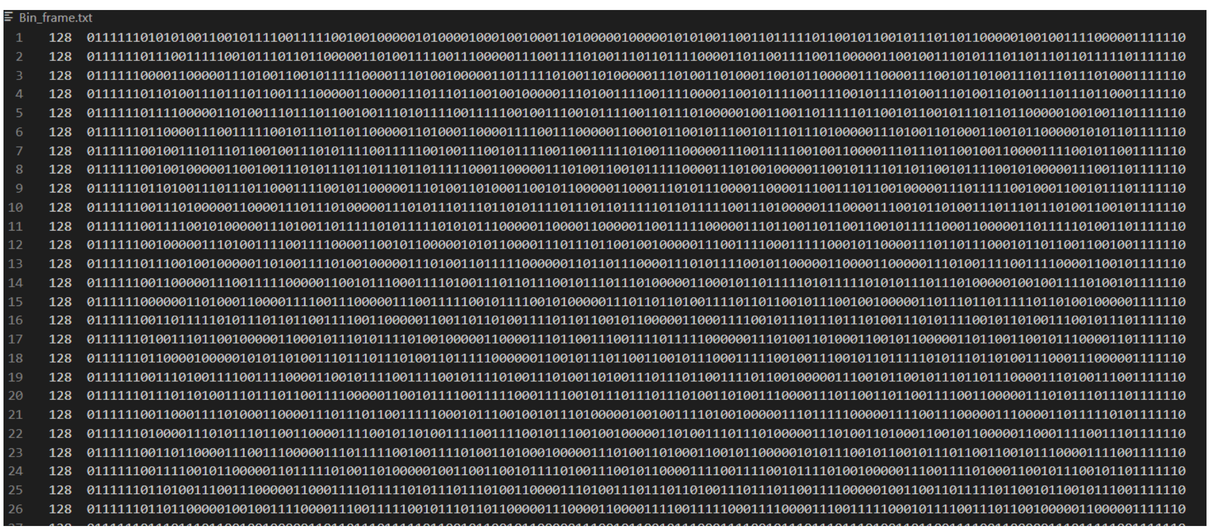
Output:

Char_frame.txt

```
1024  Test PDF
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the
industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type
and scrambled it to make a type specimen book. It has sur vived not only five centuries, but also the lea
into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the
release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing
software like Aldus PageMaker including versions of Lorem Ipsum.
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the
industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type
and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap
into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the
release of Letrase


128  t sheets containing Lore m Ipsum passages, and more recently with desktop publishing
software like Aldus PageMaker including ve


512  rsions of Lorem Ipsum.
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the
industry's standard dummy text e ver since the 1500s, when an unknown printer took a galley of type
and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap
into electronic typesetting, remaining essentially unchanged. It was popularised in  the 1960s with the
release of Letraset sheets containing Lorem Ipsum passages, and more rece


1024  ntly with desktop publishing
software like Aldus PageMaker including versions of Lorem Ipsum.
Lorem Ipsum is simply dummy text of the printing and typesetting indus try. Lorem Ipsum has been the
```

```
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the
industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type
and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap
into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the
release of Letrase


128  t sheets containing Lore m Ipsum passages, and more recently with desktop publishing
software like Aldus PageMaker including ve


512  rsions of Lorem Ipsum.
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the
industry's standard dummy text e ver since the 1500s, when an unknown printer took a galley of type
and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap
into electronic typesetting, remaining essentially unchanged. It was popularised in  the 1960s with the
release of Letraset sheets containing Lorem Ipsum passages, and more rece


1024  ntly with desktop publishing
software like Aldus PageMaker including versions of Lorem Ipsum.
Lorem Ipsum is simply dummy text of the printing and typesetting indus try. Lorem Ipsum has been the
industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type
and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap
into electronic types etting, remaining essentially unchanged. It was popularised in the 1960s with the
release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing
software like Aldus PageMaker including versions of Lorem Ipsum.
```

```
Enter the size of frames for bit stuffing:7
```



SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

Department of Computer Engineering

Class: S.Y. B.Tech. **Semester: IV**

Course Code: DJ19CEL405 **Course Name: Computer Networks Lab**

Bin_frame.txt

```
36  128  01111110111001011110100111010011010011101110110011111000001101001110111011001001110101111001111100100111001011110011011101000001001100011001111110
37  128  01111110111110110010110010111011011000001001001111000011100111110010111011011000001101000110000111100110000110001011001011100101110110111001111110
38  128  01111110010000011101001101000110010110000010101101001110111011001001110101111001111001001110010111100110011110101111001111001000011100111110010111110
39  128  01111110001100001110111011001001100001111001011001001000001100100111010111011011111000110000011101001100101111100000111010011001011111000011101001000001011111110
40  128  01111110100101111011011001011110010100000111001111010011101110001111001011000001110100110100011001011000001100011101011100001100001101111110
41  128  01111110001110110010000011101111100100011001011101110100000110000111011101000001110101110110111011101111011101111011011110011101000000111110
42  128  01111110111000011100101101001110111011101001100101111001010000011101001101111101011111010101110000011000011000001100111110000011101100110111110
43  128  01111110011001100101111100011000001101111001101000001110100111100011100001100101100000101011000001011000001100111110000011100111100001101111110
44  128  01111110100010110000011101101110001011011001100101110010010000011010011101001000001110100110111110000001101101110000011101011100001100101111110
45  128  01111110011000011000001110100111001111000011001011000001110011111000001100101110001111010011101011100101110111010000011000101101111010111110
46  128  01111110011111010101111011101100000100100111010001000001101000111000011100001100101111001111001011110011010111000011100101110111010011010000111110
47  128  01111110010000011011011011111011010010000011011111010111011011001111001100000110011011010011110110100101110111001011000001100011110010111011101111110
48  128  01111110010011101011110010110100111001011100111001101000001100010111010111010100000110000111011001110011111011110000001110100110100010111110
49  128  01111110100101100000110110011001011100001111000010000010101101001110111011101001101111000000110010111011001100101110001111010010011001001111110
50  128  01111110110111110101111010011100011100000111010011110011111000011001011110011111000001101111100100111101001101000100000111010011010001100101100000101011000111110
51  128  01111110010110010111011011100001110100111011101100111011101100111100000110010111001111000111001011101110111010011010011100001110110011111110
52  128  01111110011011001111001100000111010111011100011110100011000011101110111001111000101110010010111010000010010011110100100000111011111000111110
53  128  01111110000111100111000001110000110111110110000111010111011000111101111001011010011110011110010111001001000011010011101110100000111010001111110
54  128  01111110110100011001011000001100011110011101101100001110011100000111011111001001111010011101000100000111010011010001100101100000101011000111110
55  128  01111110101100101110110011001011100001111001111001011000001101111010011010000010011001100101111010011100101100001110011110010111010010111110
56  128  01111110000001110011110100011001011100101111010011100111000001100011110111101011011101001100001110100111011101100101110111010011100111000001111110
57  128  01111110001001100111011111011001011001011000001101101100000100100111100001110011111001011101101100000111000011000011110011111000111000010111110
58  128  01111110110011111000101111001110110010000011000011101110110010010000011011011011111011001011001100000111001011001011100011110010111010111110
59  128  01111110111011010011011001111001100000111011111001001111010011010001000001100100110010111100111101011110011001101111011110000100000111000011110
60  128  01111110011101011100010110110011010011101111001011000010000011101100110010011101110110011110000010101110011110111110100111011010011101111000001111001011001111110
61  128  01111110010110000011011001101001110101111001011000001000001110110011001001110101111001110000010100001100001110011111000101100110111000010111110
```

```
99  128  01111110110000011010011101101100100111010111001111000011101001110010111001101110100000100110011111101001101001110110111000001001001111110
100 128  01111110011110000111001111100101110110110000011010001100001111001110000011000101100101110010111010100001110100110100011001011000001010111110
101 128  01111110011010011101110110010011101011111001111001001110001011110010011110000011100111100100111011101100101100001111001001100001111001001111110
102 128  01111110110010010000011001001110101111011011011011111000110000111010011001011111000011101001000001100101110110110010111100101000001110011111110
103 128  01111110011110100111011101100011110010110000011101001101000110010110000011000111010111000011100001100111011001000001110111110010001100100111110
104 128  01111110111011101000001100001110111010000011101011101110101111011011111010111011101111000011101011100101110111100101000011100001110100111010011110
105 128  01111110001011110010100000111010011011111010111101010111000001100001100001110011111000001110110011011001011111000110000011011111010011111110
106 128  01111110011010000011101001111001111000011001010000010101100001110111011001001000001110011110001111100010111000011101101110001011010011001111110
107 128  01111110010111001001000001101001111010010000011101001101111100000011101011100001110101111001011000001100001110000111101001111001111000011101111110
108 128  01111110001011000001110011111000001100101110001111010011101101110010111011101000001100010110111110101111101010111011010000010010011110101111110
109 128  01111110001000001101000110000111100111000001110011111001011110010111011010110100111101101100101110010010000011011101101111101101001000001101111110
110 128  01111110011111010111011010011111001100000110011011010001111011000011000011110010111011011010011100111011010111011010011101011110100110001111001011110001111110
111 128  01111110111011001000001100010111010111010010000011000011101100111001111011111000000111010011010001100101100000111011001101011100001111001111110
112 128  01111110000100000101011010011101110111010011011111000000110010111010011100101110001111000001110100110100011001011000011101100110100111000011101111110
113 128  01111110101001111001110000110010111001110000011001011110100111010011101110011100111011001000011100101100101110111010011100011100000110111110
114 128  01111110110111011010011011101110011110000011001011110011111100011101110111010011010011100111011001000011100101100101110111001100001110101110011110
115 128  01111110111011000111101000110000111011101100111110001011100100101110100000100100111010010000011101111000001110011100000111000011011110111110
116 128  01111110101100001110101110110011000011110010110100111100111100101110010010000011010011101110100000111010011010001100101100000110001111000111110
117 128  01111110111011010000111001110000011101111100100111010011010001000001110100110100011001011000001010111001011010111011001100101110000011011110
118 128  01111110110011110010110000011011111010011010000010011001100101110100111001011000011100111100101111010010000011100111010001100101110010111110
119 128  01111110011110100111001110000011000111101111010111011101001110000110100111011101101011001110111011001111000010011001101111101100101100101111110
120 128  01111110110110110000010010011110000111001111100101110110110000011100001110001111001111000011100101110111010011010011100111000010111001111110
121 128  01111110000111011101100100100000110110111011111011001011001011000001110010110010111000111100101110111011101001101100111100100000111011111110
122 128  01111110110010011101001101000100000110010011001011100111101011111001001101111101100001000001110001110101110001011010011101001111001110111110
123 128  01111110101000110100111011011001111000001010111001111011111010011101110111101001111100000111001011001011000010111001101001111001110111110
124 128  01111110110000010000011101100110010011101011110011100000101000011000011100111110001011001101110000111010111001011110010100000110100111001111110
125 128  01111110111011000111101100111010111001001101001110111011001111000011101101100101111001011100111010011011111010111011001110000011011101111110
126 128  01111110110100110100000100110011011111011001010110011011101100000010010011110000111001111100101110110111011010000010000010000000000000000001111110
127 126  16111
```

Conclusion:

Different framing methods are implemented.

Department of Computer Engineering

Class: S.Y. B.Tech. **Semester: IV**

Course Code: DJ19CEL405 **Course Name: Computer Networks Lab**

Name:Kushal Patel **SAP ID:60004210058**

Date of Performance: 09-03-2023 **Date of Submission: 09-03-2023**

Experiment No: 4 (A)

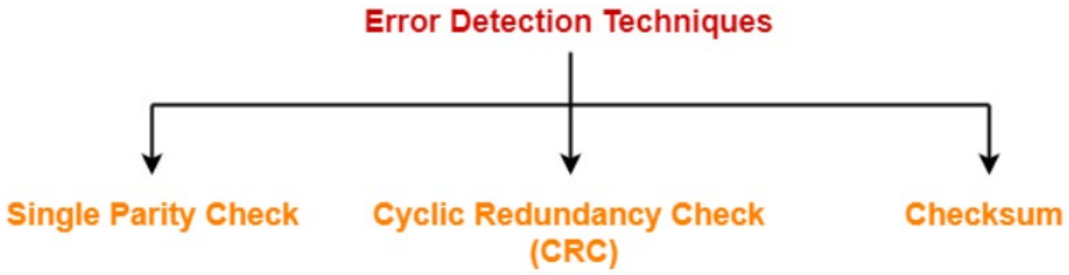
Aim: To study CRC Implemetation.

Theory:

Error Detection in Computer Networks-

Error detection is a technique that is used to check if any error occurred in the data during the transmission.

Some popular error detection methods are-



Cyclic Redundancy Check-

- Cyclic Redundancy Check (CRC) is an error detection method.
- It is based on binary division.

CRC Generator-

- CRC generator is an algebraic polynomial represented as a bit pattern.
- Bit pattern is obtained from the CRC generator using the following rule-

The power of each term gives the position of the bit and the coefficient gives the value of the bit.

Calculation Of CRC At Sender Side-

At sender side,

- A string of n 0's is appended to the data unit to be transmitted.
- Here, n is one less than the number of bits in CRC generator.
- Binary division is performed of the resultant string with the CRC generator.
- After division, the remainder so obtained is called as **CRC**.
- It may be noted that CRC also consists of n bits.

Step-02: Appending CRC To Data Unit-

At sender side,

- The CRC is obtained after the binary division.
- The string of n 0's appended to the data unit earlier is replaced by the CRC remainder.

Step-03: Transmission To Receiver-

- The newly formed code word (Original data + CRC) is transmitted to the receiver.

Step-04: Checking at Receiver Side-

At receiver side,

- The transmitted code word is received.
- The received code word is divided with the same CRC generator.
- On division, the remainder so obtained is checked.

The following two cases are possible-

Case-01: Remainder = 0

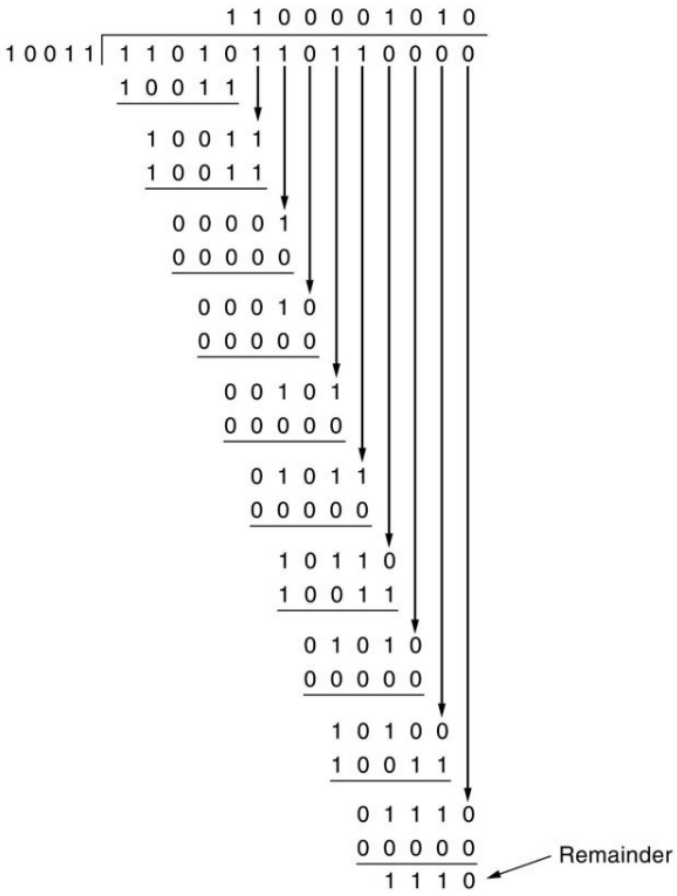
If the remainder is zero,

- Receiver assumes that no error occurred in the data during the transmission.
- Receiver accepts the data.

Case-02: Remainder ≠ 0

If the remainder is non-zero,

- Receiver assumes that some error occurred in the data during the transmission.
- Receiver rejects the data and asks the sender for retransmission.



Code:

```
def xor(a, b):
    result = []
    for i in range(1, len(b)):
        if a[i] == b[i]:
            result.append('0')
        else:
            result.append('1')
    return ''.join(result)
def flip(c):
    return '1' if (c == '0') else '0'
def CrcGen(poly, deg, bitStream, flag):
    if flag == 0:
        for i in range(0,int(deg)):
            bitStream += '0'
    dividend = bitStream
    divisor = poly
    size = len(divisor)
    temp = dividend[0:size]
    quotient = ''
    while size < len(dividend):
        while len(temp) < len(divisor):
            # print(str(temp)+' '+str(size))
            temp += dividend[size:size+1]
            size += 1
        if temp >= divisor:
            quotient += '1'
        else:
            quotient += '0'
        temp = xor(temp, divisor)
    ones = ""

    for i in range(len(temp)):
        ones += flip(temp[i])
    return ones,quotient


if __name__ == '__main__':
    deg = input("Enter the degree: ")
    poly = input("Enter the polynomial powers: ")
    bitStream = input("Enter the bit stream: ")
```

```
    rem, quot = CrcGen(poly, deg, bitStream, 0)
    print((bitStream+rem))
    rem1, quot1 = CrcGen(poly, deg, (bitStream+rem), 1)
    if rem1 =='0000':
        print("No error")
    else:
        print(rem1)
```

Output:

```
C:\Users\djsce.student\Desktop\60004210062>C:/Users/djsce.student/AppData/Local/Microsoft/WindowsApps/python3.10.exe c:/Users/djsce.student/Desktop/60004210062/Exp4.py
Enter the degree: 4
Enter the polynomial powers: 10011
Enter the bit stream: 1101011011
11010110111110
No error

C:\Users\djsce.student\Desktop\60004210062>C:/Users/djsce.student/AppData/Local/Microsoft/WindowsApps/python3.10.exe c:/Users/djsce.student/Desktop/60004210062/Exp4.py
Enter the degree: 4
Enter the polynomial powers: 10011
Enter the bit stream: 11010110110000
11010110110001110
No error

C:\Users\djsce.student\Desktop\60004210062>
```

Conclusion:

Thus, we have implemented CRC.

Department of Computer Engineering

Class: S.Y. B.Tech. **Semester: IV**

Course Code: DJ19CEL405 **Course Name: Computer Networks Lab**

Name: Kushal Patel **SAP ID:60004210058**

Experiment No: 4 (B)

Aim: To study Hamming code.

Theory:

General Algorithm of Hamming code: Hamming Code is simply the use of extra parity bits to allow the identification of an error.

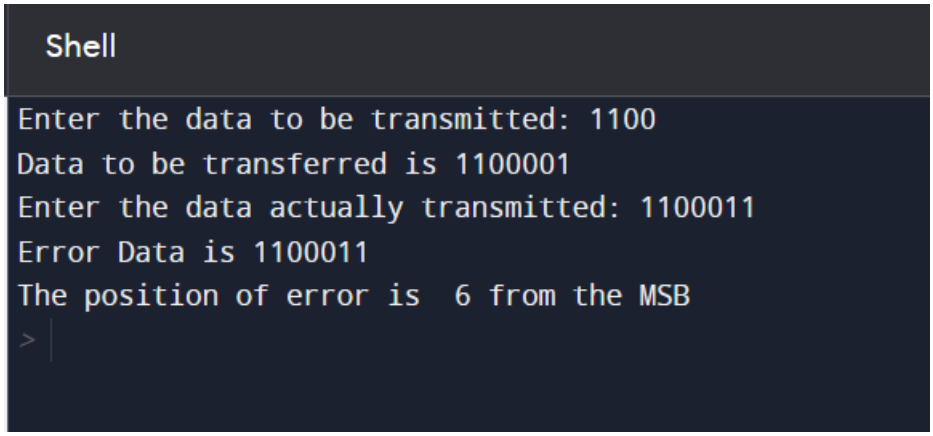
- Write the bit positions starting from 1 in binary form (1, 10, 11, 100, etc).
- All the bit positions that are a power of 2 are marked as parity bits (1, 2, 4, 8, etc).
- All the other bit positions are marked as data bits.
- Each data bit is included in a unique set of parity bits, as determined its bit position in binary form. a. Parity bit 1 covers all the bits positions whose binary representation includes a 1 in the least significant position (1, 3, 5, 7, 9, 11, etc). b. Parity bit 2 covers all the bits positions whose binary representation includes a 1 in the second position from the least significant bit (2, 3, 6, 7, 10, 11, etc). c. Parity bit 4 covers all the bits positions whose binary representation includes a 1 in the third position from the least significant bit (4–7, 12–15, 20–23, etc). d. Parity bit 8 covers all the bits positions whose binary representation includes a 1 in the fourth position from the least significant bit bits (8–15, 24–31, 40–47, etc). e. In general, each parity bit covers all bits where the bitwise AND of the parity position and the bit position is non-zero.
- Since we check for even parity set a parity bit to 1 if the total number of ones in the positions it checks is odd.
- Set a parity bit to 0 if the total number of ones in the positions it checks is even.

Code:

```
def calcRedundantBits(length):
    for i in range(length):
        if(2**i >= length+i+1):
            return i

def calcParityBits(arr, red):
    n = len(arr)

    for i in range(red):
        val =0

        for j in range(1, n+1):
            if(j & (2**i) == (2**i)):
                val = val ^ int(arr[-1*j])

        arr = arr[:n-(2**i)] + str(val) + arr[n-(2**i)+1:]

    return arr

def posRedundantBits(data, red):
    j=0
    k=1
    length=len(data)
    res =''

    for i in range(1, length+red+1):
        if(i==2**j):
            res = res + '0'
            j += 1
        else:
            res= res + data[-1*k]
            k +=1

    return res[::-1]

def detectError(arr, nr):
    n = len(arr)
    res=0

    for i in range(nr):
        val = 0
        for j in range(1, n + 1):
            if(j & (2**i) == (2**i)):
                val = val ^ int(arr[-1 * j])

        res = res + val*(10**i)
```

```
    return int(str(res), 2)

data = input("Enter the data to be transmitted: ")
length = len(data)
red = calcRedundantBits(length)
arr = posRedundantBits(data, red)
arr = calcParityBits(arr, red)
print("Data to be transferred is " + arr)
arr = input("Enter the data actually transmitted: ")
print("Error Data is " + arr)
correction = detectError(arr, red)
if(correction==0):
    print("There is no error in the received message.")
else:
    print("The position of error is ",len(arr)-correction+1,"from the MSB")
```

Output:



Conclusion:

Thus, we have implemented Hamming code error detection and correction.

Name: Kushal Patel **SAP ID: 60004210058**

EXPERIMENT NO.5

AIM:

Implement Dikstra's and Bellman Ford algorithm to find shortest path in the network. (Routing Algorithm)

THEORY:

The Dijkstra's algorithm finds the shortest path from a particular node, called the source node to every other node in a connected graph. It produces a shortest path tree with the source node as the root. It is profoundly used in computer networks to generate optimal routes with the aim of minimizing routing costs.
This algorithm takes as input a directed weighted graph and a starting vertex. It produces all the shortest paths from the starting vertex to all other vertices.

Now let's describe the notation that we used in the pseudocode. The first step is to initialize the vertices. The algorithm initially set the distance from starting vertex to all other vertices to infinity. The distance between starting vertex to itself is 0. The variable D[] denotes the distances in this algorithm.

After the initialization step, the algorithm started calculating the shortest distance from the starting vertex to all other vertices. This step runs (|V| - 1) times. Within this step, the algorithm tries to explore different paths to reach other vertices and calculates the distances. If the algorithm finds any distance of a vertex that is smaller then the previously stored value then it relaxes the edge and stores the new value.

Finally, when the algorithm iterates (|V| - 1) times and relaxes all the required edges, the algorithm gives a last check to find out if there is any negative cycle in the graph.

If there exists a negative cycle then the distances will keep decreasing. In such a case, the algorithm terminates and gives an output that the graph contains a negative cycle hence the algorithm can't compute the shortest path. If there is no negative cycle found, the algorithm returns the shortest distances.

The Bellman-Ford algorithm is an example of Dynamic Programming. It starts with a starting vertex and calculates the distances of other vertices which can be reached by one edge. It then continues to find a path with two edges and so on. The Bellman-Ford algorithm follows the bottom-up approach.

CODE:

i)Dijkstra:

```
// Dijkstra
#include <limits.h>
#include <stdio.h>
#include <stdbool.h>
#define V 100
int minDistance(int dist[], int vis[], int n)
{
int min = INT_MAX, min_index;
for (int i = 0; i < n; i++)
if (vis[i] == 0 && dist[i] <= min) min = dist[i], min_index = i;
return min_index;
}
void printSolution(int dist[], int n)
{
printf("Vertex \t\t Distance from Source\n"); for (int i = 0; i < n; i++)
printf("%d \t\t %d\n", i, dist[i]);
}
void dijkstra(int graph[V][V], int src, int n)
{
int dist[n];
int vis[n];
for (int i = 0; i < n; i++)
dist[i] = INT_MAX, vis[i] = 0; dist[src] = 0;
for (int count = 0; count < n - 1; count++)
{
int u = minDistance(dist, vis, n); vis[u] = 1;
for (int v = 0; v < n; v++)
if (!vis[v] && graph[u][v] && dist[u] != INT_MAX && dist[u] + graph[u][v]
<dist[v])
dist[v] = dist[u] + graph[u][v];
}
printSolution(dist, n);
}
void main()
{
int graph[V][V]; int n, i, j;
printf("Enter number of nodes: ");
scanf("%d", &n);
for (i = 0; i < n; i++)
{
printf("Enter distance from node %d:\n", (i + 1));
 for (j = 0; j < n; j++)
{
printf("Node %d: ", j + 1);
scanf("%d", &graph[i][j]);
}
}
dijkstra(graph, 0, n);
}
```

ii) Bellmond ford:

```
#include<stdio.h>
int dist[10], cost[10][10], path[10], n, v, x;
int main()
{
    int i, j, u, v, c;
    printf("\nEnter number of vertices: ");
    scanf("%d", &n);
    printf("\nEnter cost matrix: \n");
    for(i=1;i<=n;i++)
    {
        for(j=1;j<=n;j++)
        {
            scanf("%d", &cost[i][j]);
        }
    }
    printf("\nEnter source vertex: ");
    scanf("%d", &v);
    for(i=1;i<=n;i++)
    {
        if(i==v)
        {
            dist[i]=0;}
        else
        {
            dist[i]=100;
        }
        path[i]=NULL;
    }
    for(i=1; i<n; i++)
    {
        for(u=1; u<=n; u++)
        {
            for(v=1;v<=n;v++)
            {
                if((cost[u][v])!=100 && u!=v)
                {
                    if((dist[u]+cost[u][v]) < dist[v])
                    {
                        dist[v]=dist[u]+cost[u][v];
                        path[v]=u;
                    }
                }
            }
        }
    }
    printf("\n\nDistance\tPath\n");
    for(i=1;i<=n;i++)
```

```
    {
        printf("\ndist[%d]=%d\tp[%d]=%d\n", i, dist[i], i, path[i]);
    }
    return 0;
}
```

OUTPUT:

i) Dijkstra:

```
Enter number of nodes: 3
Enter distance from node 1:
Node 1: 10
Node 2: 20
Node 3: 40
Enter distance from node 2:
Node 1: 8
Node 2: 7
Node 3: 9
Enter distance from node 3:
Node 1: 60
Node 2: 5
Node 3: 4
Vertex            Distance from Source
0                 0
1                 20
2                 29
PS D:\OneDrive\Desktop\SEM IV\CN\PRAC CODES\output>
```

ii) Bellmond ford:

```
Enter number of vertices: 5

Enter cost matrix:
0 4 2 0 0
0 0 3 2 3
0 1 0 4 5
0 0 0 0 0
0 0 0 -5 0

Enter source vertex: 1


Distance        Path

dist[1]=-15     p[1]=4

dist[2]=-15     p[2]=4

dist[3]=-15     p[3]=4

dist[4]=-20     p[4]=5

dist[5]=-15     p[5]=4
```

CONCLUSION: Thus, we have implemented Dijkstra and Bellman Ford Algorithm using C.

Experiment No: 06

Name: Kushal Patel SAP ID: 60004210058

Aim: Write a program to identify the class and subnet address of the given IP address.

Theory:

Internet addresses are allocated by the InterNIC, the organization that administers the Internet.

These IP addresses are divided into classes. The most common of them are classes A, B, and C. Classes D and E exist, but aren't used by end users. Each of the address classes has a different default subnet mask. You can identify the class of an IP address by looking at its first octet. Following are the ranges of Class A, B, and C Internet addresses, each with an example address:

- Class A networks use a default subnet mask of 255.0.0.0 and have 0-127 as their first octet. The address 10.52.36.11 is a class A address. Its first octet is 10, which is between 1 and 126, inclusive.
- Class B networks use a default subnet mask of 255.255.0.0 and have 128-191 as their first octet. The address 172.16.52.63 is a class B address. Its first octet is 172, which is between 128 and 191, inclusive.
- Class C networks use a default subnet mask of 255.255.255.0 and have 192-223 as their first octet. The address 192.168.123.132 is a class C address. Its first octet is 192, which is between 192 and 223, inclusive.

In some scenarios, the default subnet mask values don't fit the organization needs for one of the following reasons:

- The physical topology of the network
- The numbers of networks (or hosts) don't fit within the default subnet mask restrictions Subnetting:
- A Class A, B, or C TCP/IP network can be further divided, or sub netted, by a system administrator. It becomes necessary as you reconcile the logical address scheme of the Internet (the abstract world of IP addresses and subnets) with the physical networks in use by the real world.
- A system administrator who is allocated a block of IP addresses may be administering networks that aren't organized in a way that easily fits these addresses. For example, you have a wide area network with 150 hosts on three networks (in different cities) that are connected by a TCP/IP router. Each of these three networks has 50 hosts. You are allocated the class C network 192.168.123.0. (For illustration, this address is actually from a range that isn't allocated on the Internet.) It means that you can use the addresses 192.168.123.1 to 192.168.123.254 for your 150 hosts.
- Two addresses that can't be used in your example are 192.168.123.0 and

192.168.123.255 because binary addresses with a host portion of all ones and all zeros are invalid. The zero address is invalid because it's used to specify a network without specifying a host.
The 255 addresses (in binary notation, a host address of all ones) is used to broadcast a message to every host on a network. Just remember that the first and last address in any network or subnet can't be assigned to any individual host.

- You should now be able to give IP addresses to 254 hosts. It works fine if all 150 computers are on a single network. However, your 150 computers are on three separate physical networks. Instead of requesting more address blocks for each network, you divide your network into subnets that enable you to use one block of addresses on multiple physical networks.
- In this case, you divide your network into four subnets by using a subnet mask that makes the network address larger and the possible range of host addresses smaller. In other words, you are 'borrowing' some of the bits used for the host address, and using them for the network portion of the address. The subnet mask 255.255.255.192 gives you four networks of 62 hosts each. It works because in binary notation, 255.255.255.192 is the same as 1111111.11111111.1111111.11000000. The first two digits of the last octet become network addresses, so you get the additional networks 00000000 (0), 01000000 (64), 10000000 (128) and 11000000 (192). (Some administrators will only use two of the subnetworks using 255.255.255.192 as a subnet mask. For more information on this topic, see RFC 1878.) In these four networks, the last six binary digits can be used for host addresses.
- Using a subnet mask of 255.255.255.192, your 192.168.123.0 network then becomes the four networks 192.168.123.0, 192.168.123.64, 192.168.123.128 and 192.168.123.192. These four networks would have as valid host addresses:
- 192.168.123.1-62 192.168.123.65-126 192.168.123.129-190 192.168.123.193-254
- Remember, again, that binary host addresses with all ones or all zeros are invalid, so you can't use addresses with the last octet of 0, 63, 64, 127, 128, 191, 192, or 255. • You can see how it works by looking at two host addresses, 192.168.123.71 and 192.168.123.133. If you used the default Class C subnet mask of 255.255.255.0, both addresses are on the 192.168.123.0 network. However, if you use the subnet mask of 255.255.255.192, they are on different networks; 192.168.123.71 is on the 192.168.123.64 network, 192.168.123.133 is on the 192.168.123.128 network.

Code:

```
import java.io.*;
// import java.net.InetAddress;

public class subnet{
   public static void main(String[] args) throws IOException{
      System.out.println("Enter IP Address: ");
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
String ip = br.readLine();          String checkclass = ip.substring(0,3);          int cc =
Integer.parseInt(checkclass);
String mask = null;         if(cc>0 && cc<224){
if(cc<=127){
System.out.println("IP Address is of
```

```
Class A");
mask="255.0.0.0";
        }
        if(cc>=128 && cc<=191){
           System.out.println("IP Address is of Class B");
mask="255.255.0.0";
        }
        if(cc>=192 && cc<=223){
           System.out.println("IP Address is of Class C");
mask="255.255.255.0";
        }
        if(cc>=224){
           System.out.println("IP Address is used for Multicasting or reserved");
        }
      }
      System.out.println("Subnet Mask:\n"+mask);
      String networkAddr="";

      String[] ipAddrParts = ip.split("\\.");
      String[] maskParts = mask.split("\\.");

      for(int i=0;i<=3;i++){          int x =
Integer.parseInt(ipAddrParts[i]);          int y
= Integer.parseInt(maskParts[i]);

         int z = x & y;

         networkAddr += z+".";
      }
      System.out.println("SUBNET Address:\n"+networkAddr);
   }
}
```

Output:

```
Enter IP Address:
187.32.13.12
IP Address is of Class B
Subnet Mask:
255.255.0.0
SUBNET Address:
187.32.0.0.
```

Conclusion:

The program to identify the class and subnet address of the given IP address is implemented.

Department of Computer Engineering

Class: S.Y. B.Tech. **Semester: IV**

Course Code: DJ19CEL405 **Course Name: Computer Networks Lab**

Name: Kushal Patel **SAP ID:60004210058**

Date of Performance: 06-04-2023 **Date of Submission: 13-04-2023**

Experiment No: 9

Aim: Write a program for implementing voting system using Client Server Model. Assume minimum 3 contestant and 6 voters. Client system can vote in favor of contestant. Result will be computed and displayed by Server.

Theory:

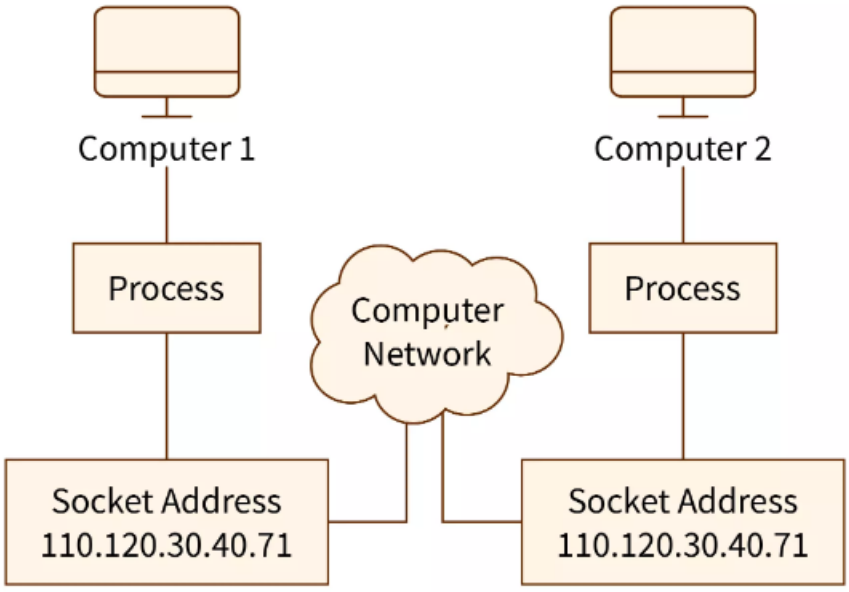
Sockets allow communication of two processes that are running on the same or different machines. Sockets are the end of two-way communication between two programs that are running on the networks.

Sockets are mostly used in client-server architecture for communication between multiple applications.

Socket programming tells us how we can use socket API for creating communication between local and remote processes.

The socket is created by the combination of the IP address and port number of the software. With this combination, the process knows the system address and address of the application where data is to be sent.

: is used to separate IP address and port number. For
eg: 192.168.1.67:80, 155.2.12.23:77, etc.



Code:

CLIENT:

```
import socket
def Main():
    host = '127.0.0.1'
    port = 12345
    s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    s.connect((host,port))
    message = "hello"
    while True:
        s.send(message.encode('ascii'))
        data = s.recv(1024)
        print('Received from the server :',str(data.decode('ascii')))
        message = input("Enter your vote:")
        ans = input('Do you want to continue(y/n) :')
```

```
    if ans == 'y':
        continue
    else:
        message = 'finish'
        s.send(message.encode('ascii'))
        break
  s.close()

if __name__ == '__main__':
  Main()
```

SERVER:

```
import socket
from _thread import *
import threading
# print_lock = threading.Lock()
def threaded(c, votes):
  while True:
    data = c.recv(1024)
    if not data:
      print('Bye')
      # print_lock.release()
      break
    if str(data.decode('ascii')) == '1':
      votes[0] += 1
    if str(data.decode('ascii')) == '2':
      votes[1] += 1
```

```
        if str(data.decode('ascii')) == '3':
            votes[2] += 1
        if str(data.decode('ascii')) == 'finish':
            imax= votes[0]
            for i in votes:
                if i>imax:
                    imax = i
            print("Maximum votes are for:", chr(64+int(imax)))
            # print_lock.release()
            c.close()
            break
        data = "if you want to vote for A send 1, for B send 2 or for C send 3"
        c.send(data.encode('ascii'))
        for i in votes:
            print(i)
    c.close()


def Main():
    host = ""
    port = 12345
    votes = [0,0,0]


    try:
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        print ("Socket successfully created")
    except socket.error as err:
        print ("socket creation failed with error %s" %(err))
```

```
    s.bind((host, port))
    print("socket binded to port", port)
    s.listen(5)
    print("socket is listening")
    while True:
        c, addr = s.accept()
        # print_lock.acquire()
        print('Connected to :', addr[0], ':', addr[1])
        try:
            start_new_thread(threaded, (c,votes))
        except:
            print ("Error: unable to start thread")
    s.close()

if __name__ == '__main__':
    Main()
```

Output:

```
Received from the server : if you want to vote for A send 1, for B send 2 or for C send 3
Enter your vote:1
Do you want to continue(y/n) :y
Received from the server : if you want to vote for A send 1, for B send 2 or for C send 3
Enter your vote:2
Do you want to continue(y/n) :y
Received from the server : if you want to vote for A send 1, for B send 2 or for C send 3
Enter your vote:1
Do you want to continue(y/n) :n
```

```
Socket successfully created
socket binded to port 12345
socket is listening
Connected to : 127.0.0.1 : 51670
0
0
0
1
0
0
1
1
0
Maximum votes are for: A
```

Conclusion:

Thus, we have implemented socket programming in client – server voting system.

Experiment No: 10

Name: Kushal Patel SAP ID: 60004210058

Aim: Write a program to implement routing protocol.

Theory

A routing protocol specifies how routers communicate with each other, disseminating information that enables them to select routes between any two nodes on a computer network. Routing algorithms determine the specific choice of route. Each router has a priori knowledge only of networks attached to it directly.

Types of Routing Protocols

Table 3-1 showed how routing protocols can be classified according to various characteristics. This section gives an overview of the most common IP routing protocols. Most of these routing protocols will be examined in detail in other chapters. For now, this section gives a very brief overview of each protocol.

Classifying Routing Protocols

Routing protocols can be classified into different groups according to their characteristics. Specifically, routing protocols can be classified by their:

- Purpose: Interior Gateway Protocol (IGP) or Exterior Gateway Protocol (EGP)
- Operation: Distance vector protocol, link-state protocol, or path-vector protocol
- Behavior: Classful (legacy) or classless protocol

For example, IPv4 routing protocols are classified as follows:

- RIPv 1 (legacy): IGP, distance vector, classful protocol
- IGRP (legacy): IGP, distance vector, classful protocol developed by Cisco (deprecated from 12.2 IOS and later)
- RIPv2: IGP, distance vector, classless protocol
- EIGRP: IGP, distance vector, classless protocol developed by Cisco
- OSPF: IGP, link-state, classless protocol • IS-IS: IGP, link-state, classless protocol
- BGP: EGP, path-vector, classless protocol

The classful routing protocols, RIPv 1 and IGRP, are legacy protocols and are only used in older networks. These routing protocols have evolved into the classless routing protocols, RIPv2 and EIGRP, respectively. Link-state routing protocols are classless by nature.

Figure 3-9 displays a hierarchical view of dynamic routing protocol classification.

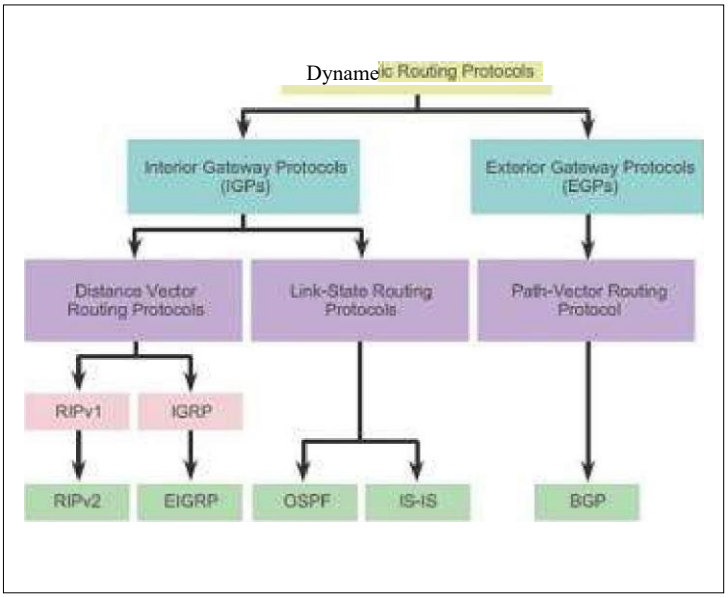


Figure 3-9 Routing Protocol Classification

IGP and EGP Routing Protocols (3.1.4.2)

An autonomous system (AS) is a collection of routers under a common administration such as a company or an organization. An AS is also known as a routing domain. Typical examples of an AS are a company's internal network and an ISP's network.

The Internet is based on the AS concept; therefore, two types of routing protocols are required:

- Interior Gateway Protocols (IGP): Used for routing within an AS. It is also referred to as intra-AS routing. Companies, organizations, and even service providers use an IGP on their internal networks. IGPs include RIP, EIGRP, OSPF, and IS-IS.
- Exterior Gateway Protocols (EGP): Used for routing between autonomous systems. It is also referred to as inter-AS routing. Service providers and large companies may interconnect using an EGP. The Border Gateway Protocol (BGP) is the only currently viable EGP and is the official routing protocol used by the Internet.

NOTE

Because BGP is the only EGP available, the term EGP is rarely used; instead, most engineers simply refer to BGP.

The example in Figure 3-10 provides simple scenarios highlighting the deployment of IGPs, BGP, and static routing.

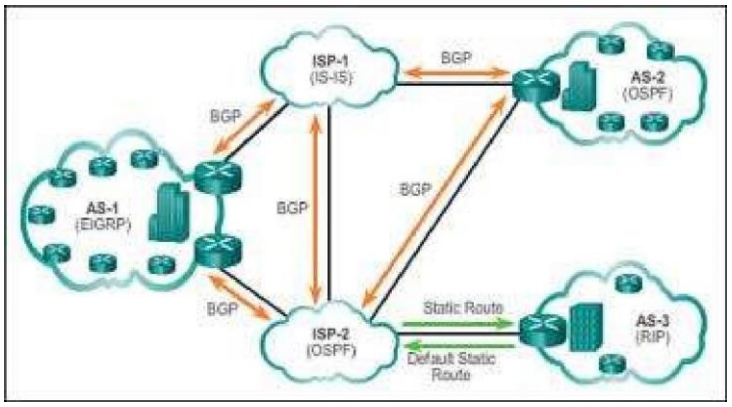


Figure 3-10 IGP versus EGP Routing Protocols

There are five individual autonomous systems in the scenario:

- ISP-I: This is an AS and it uses IS-IS as the IGP. It interconnects with other autonomous systems and service providers using BGP to explicitly control how traffic is routed.
- ISP-2: This is an AS and it uses OSPF as the IGP. It interconnects with other autonomous systems and service providers using BGP to explicitly control how traffic is routed.
- AS-I: This is a large organization and it uses EIGRP as the IGP. Because it is multihomed (i.e., connects to two different service providers), it uses BGP to explicitly control how traffic enters and leaves the AS.
- AS-2: This is a medium-sized organization and it uses OSPF as the IGP. It is also multihomed; therefore, it uses BGP to explicitly control how traffic enters and leaves the
- AS-3: This is a small organization with older routers within the AS; it uses RIP as the IGP. BGP is not required because it is single-homed (i.e., connects to one service provider). Instead, static routing is implemented between the AS and the service provider.

NOTE

BGP is beyond the scope of this course and is not discussed in detail.

Distance Vector Routing Protocols (3.1.4.3)

Distance vector means that routes are advertised by providing two characteristics:

- Distance: Identifies how far it is to the destination network and is based on a metric such as the hop count, cost, bandwidth, delay, and more
- Vector: Specifies the direction of the next-hop router or exit interface to reach the destination

For example, in Figure 3-11, RI knows that the distance to reach network 172.16.3.0/24 is one hop and that the direction is out of the interface Serial 0/0/0 toward R2.

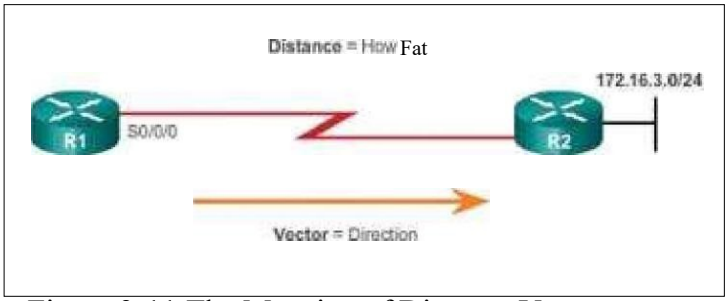


Figure 3-11 The Meaning of Distance Vector

A router using a distance vector routing protocol does not have the knowledge of the entire path to a destination network. Distance vector protocols use routers as sign posts along the path to the final destination. The only information a router knows about a remote network is the

distance or metric to reach that network and which path or interface to use to get there. Distance vector routing protocols do not have an actual map of the network topology.

There are four distance vector IPv4 IGPs:

- RIPv1: First generation legacy protocol
- RIPv2: Simple distance vector routing protocol
- IGRP: First generation Cisco proprietary protocol (obsolete and replaced by EIGRP)
- EIGRP: Advanced version of distance vector routing

Link-State Routing Protocols (3.1.4.4)

In contrast to distance vector routing protocol operation, a router configured with a link-state routing protocol can create a complete view or topology of the network by gathering information from all of the other routers.

To continue our analogy of sign posts, using a link-state routing protocol is like having a complete map of the network topology. The sign posts along the way from source to destination are not necessary, because all link-state routers are using an identical map of the network. A link-state router uses the link-state information to create a topology map and to select the best path to all destination networks in the topology.

RIP-enabled routers send periodic updates of their routing information to their neighbors. Linkstate routing protocols do not use periodic updates. After the network has converged, a link-state update is only sent when there is a change in the topology. For example, in Figure 3-12, the linkstate update is sent when the 172.16.3.0 network goes down.

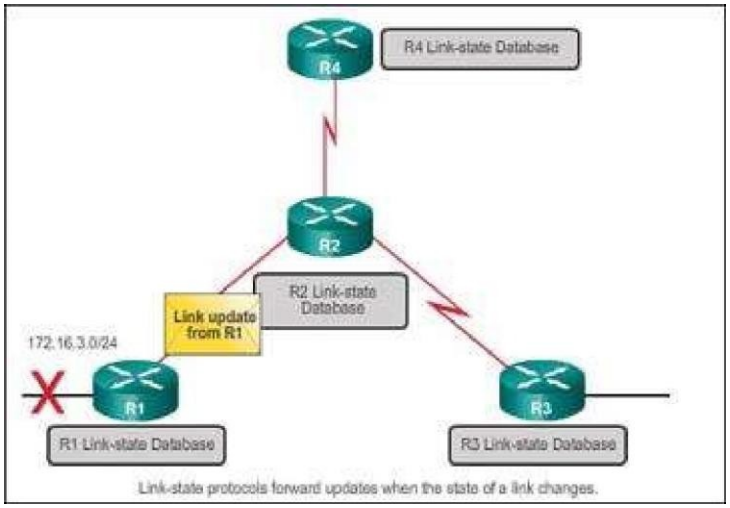


Figure 3-12 Link-State Protocol Operation

Video 3.1.4.4: Link-State Protocol Operation

Go to the online course and play the animation to see how a link-state update is only sent when the 172.16.3.0 network goes down.

Link-state protocols work best in situations where:

- The network design is hierarchical, usually occurring in large networks
- Fast convergence of the network is crucial

- The administrators have good knowledge of the implemented link-state routing protocol

There are two link-state IPv4 IGPs:

- OSPF: Popular standards-based routing protocol
- IS-IS: Popular in provider networks

Classful Routing Protocols (3.1.4.5)

The biggest distinction between classful and classless routing protocols is that classful routing protocols do not send subnet mask information in their routing updates. Classless routing protocols include subnet mask information in the routing updates.

The two original IPv4 routing protocols developed were RIPv I and IGRP. They were created when network addresses were allocated based on classes (i.e., class A, B, or C). At that time, a routing protocol did not need to include the subnet mask in the routing update, because the network mask could be determined based on the first octet of the network address.

NOTE

Only RIPv1 and IGRP are classful. All other IPv4 and IPv6 routing protocols are classless. Classful addressing has never been a part of IPv6.

The fact that RIPvl and IGRP do not include subnet mask information in their updates means that they cannot provide variable-length subnet masks (VLSMs) and Classless Inter-Domain Routing (CDR).

Classful routing protocols also create problems in discontiguous networks. A discontiguous network is when subnets from the same classful major network address are separated by a different classful network address.

To illustrate the shortcoming of classful routing, refer to the topology in Figure 3-13.

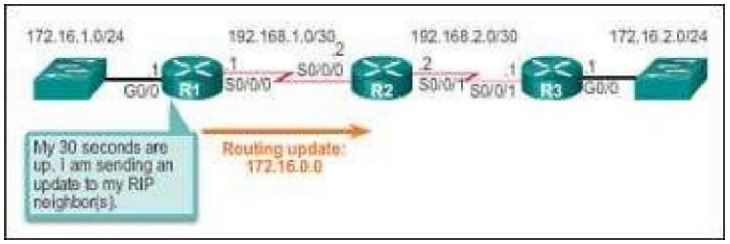


Figure 3-13 RI Forwards a Classful Update to R2

Notice that the LANs of RI (172.16.1.0/24) and R3 (172.16.2.0/24) are both subnets of the same class B network (172.16.0.0/16). They are separated by different classful network addresses (192.168.1.0/30 and 192.168.2.0/30).

When RI forwards an update to R2, RIPv 1 does not include the subnet mask information with the update; it only forwards the class B network address 172.16.0.0.

PO receives and processes the update. It then creates and adds an entry for the class B 172.16.0.0/16 network in the routing table, as shown in Figure 3-14.

Figure 3-14 R2 Adds the Entry for 172.16.0.0 via RI

When R3 forwards an update to R2, it also does not include the subnet mask information and therefore only forwards the classful network address 172.16.0.0.

R2 receives and processes the update and adds another entry for the classful network address 172.16.0.0/16 to its routing table, as shown in Figure 3-15. When there are two entries with identical metrics in the routing table, the router shares the load of the traffic equally among the two links. This is known as load balancing.

Figure 3-15 R2 Adds the Entry for 172.16.0.O via R3

Discontinuous networks have a negative impact on a network. For example, a ping to 172.16.1. I would return "U.U.U" because R2 would forward the first ping out its Serial 0/0/1 interface toward PO, and R3 would return a Destination Unreachable (U) error code to R2. The second ping would exit out ofR2's Serial 0/0/0 interface toward RI, and RI would return a successful code (.). This pattern would continue until the ping command is done.

Classless Routing Protocols (3.1.4.6)

Modern networks no longer use classful IP addressing and the subnet mask cannot be determined by the value of the first octet. The classless IPv4 routing protocols (RIPv2, EIGRP, OSPF, and ISIS) all include the subnet mask information with the network address in routing updates. Classless routing protocols support VLSM and CIDR.

IPv6 routing protocols are classless. The distinction whether a routing protocol is classful or classless typically only applies to IPv4 routing protocols. All IPv6 routing protocols are considered classless because they include the prefix-length with the IPv6 address.

Figures 3-16 through 3-18 illustrate how classless routing solves the issues created with classful routing.

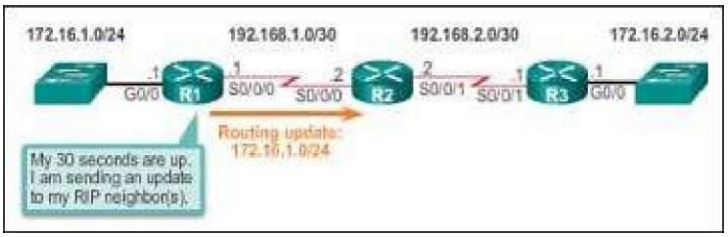


Figure 3-16 RI Forwards a Classless Update to R2


```
R2# show ip route | begin Gateway
Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 1 subnets
R       172.16.1.0 [120/1] via 192.168.1.1, 00:00:06,
        Serial0/0/0
     192.168.1.0/24 is variably subnetted, 2 subnets,
     2 masks
C       192.168.1.0/30 is directly connected, Serial0/0/0
L       192.168.1.2/32 is directly connected, Serial0/0/0
R2#
```

Figure 3-17 R2 Adds the Entry for the 172.16.1.0/24 Network via RI

```
R2# show ip route | begin Gateway
Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 2 subnets
R       172.16.1.0 [120/1] via 192.168.1.1, 00:00:03,
        Serial0/0/0
R       172.16.2.0 [120/1] via 192.168.2.1, 00:00:03,
        Serial0/0/1
     192.168.1.0/24 is variably subnetted, 2 subnets,
     2 masks
C       192.168.1.0/30 is directly connected, Serial0/0/0
L       192.168.1.2/32 is directly connected, Serial0/0/0
     192.168.2.0/24 is variably subnetted, 2 subnets,
     2 masks
C       192.168.2.0/30 is directly connected, Serial0/0/1
L       192.168.2.2/32 is directly connected, Serial0/0/1
R2#
```

Figure 3-18 Entry for the 172.16.2.0/24 Network via R3

In the discontiguous network design of Figure 3-16, the classless protocol RIPv2 has been implemented on all three routers. When RI forwards an update to R2, RIPv2 includes the subnet mask information with the update 172.16.1.0/24.

In Figure 3-17, R2 receives, processes, and adds two entries in the routing table. The first line displays the classful network address 172.16.0.0 with the /24 subnet mask of the update. This is known as the parent route. The second entry displays the VLSM network address 172.16. I .0 with the exit and next-hop address. This is referred to as the child route. Parent routes never include an exit interface or next-hop IP address.

When R3 forwards an update to R2, RIPv2 includes the subnet mask information with the update 172.16.2.0/24.

R2 receives, processes, and adds another child route entry 172.16.2.0/24 under the parent route entry 172.16.0.0, as shown in Figure 3-18.

A ping from R2 to 172.16.1.1 would now be successful.

Routing Protocol Characteristics (3.1.4.7)

Routing protocols can be compared based on the following characteristics:

- Speed of convergence: Speed of convergence defines how quickly the routers in the network topology share routing information and reach a state of consistent knowledge. The faster the convergence, the more preferable the protocol. Routing loops can occur when inconsistent routing tables are not updated due to slow convergence in a changing network.
- Scalability: Scalability defines how large a network can become, based on the routing protocol that is deployed. The larger the network is, the more scalable the routing protocol needs to be.
- Classful or classless (use of VLSM): Classful routing protocols do not include the subnet mask and cannot support variable-length subnet mask (VLSM). Classless routing protocols include the subnet mask in the updates. Classless routing protocols support VLSM and better route summarization.
- Resource usage: Resource usage includes the requirements of a routing protocol such as memory space (RAM), CPU utilization, and link bandwidth utilization. Higher resource requirements necessitate more powerful hardware to support the routing protocol operation, in addition to the packet forwarding processes.
- Implementation and maintenance: Implementation and maintenance describes the level of knowledge that is required for a network administrator to implement and maintain the network based on the routing protocol deployed.

Table 3-4 summarizes the characteristics of each routing protocol.

Table 3-4 Comparing Routing Protocols

	Distance Vector		Link-State			
	RIPv1	RIPv2	IGRP	EIGRP	OSPF	IS-IS
Speed of Convergence	Slow	Slow	Slow	Fast	Fast	Fast
Scalability — Size of Network	Small	Small	Small	Large	Large	Large
Use of VLSM	No	Yes	No	Yes	Yes	Yes
Resource Usage	Low	Low	Low	Medium	High	High
Implementation and Maintenance	Simple	Simple	Simple	Complex	Complex	Complex

Conclusion:

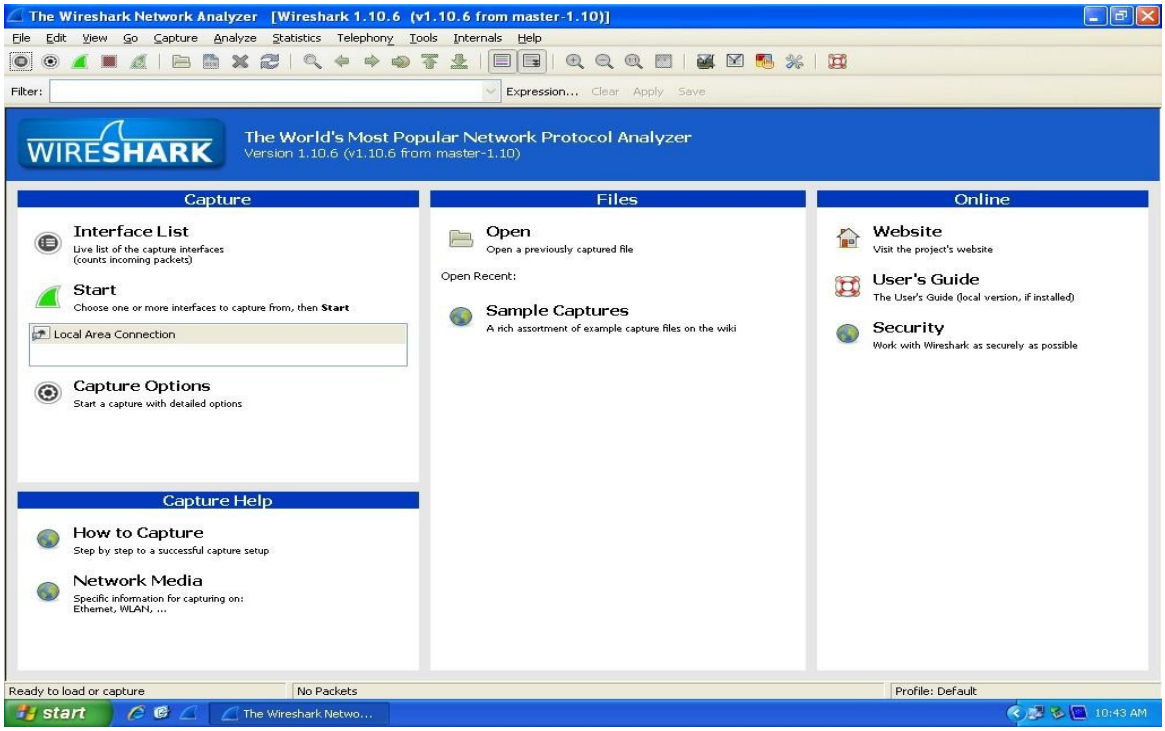
Thus we have studied and implemented routing protocol

Experiment No: 11

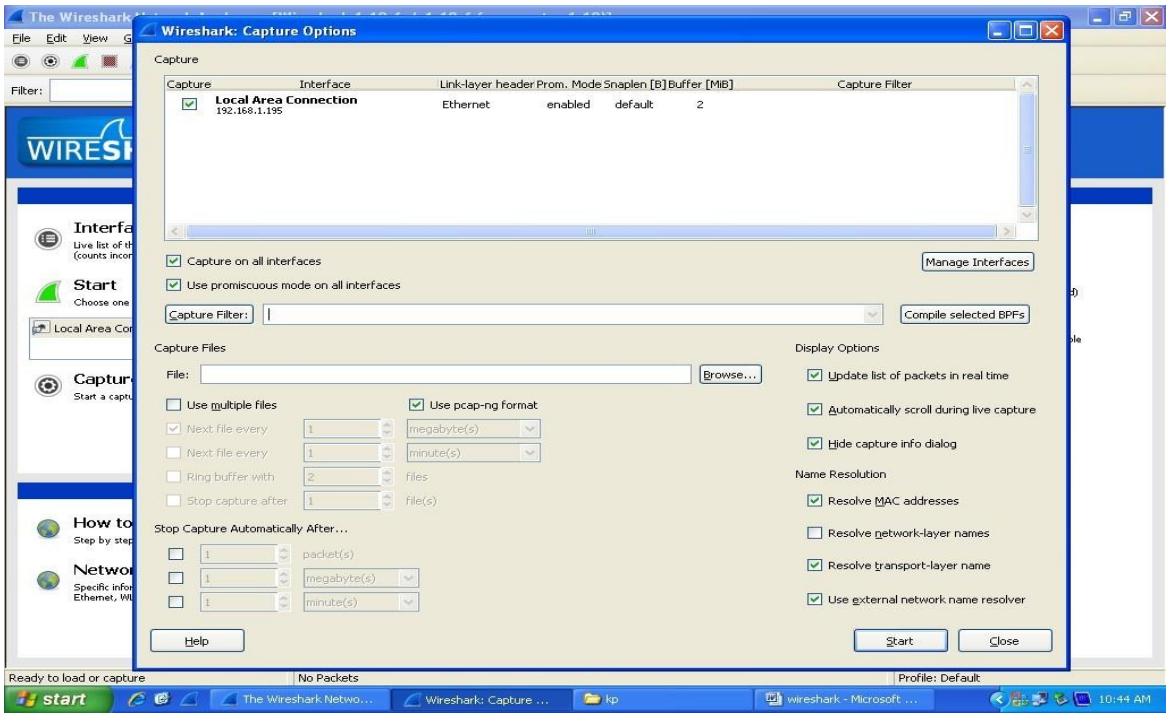
Name: Kushal Patel SAP ID: 60004210058

Aim- To study Wireshark

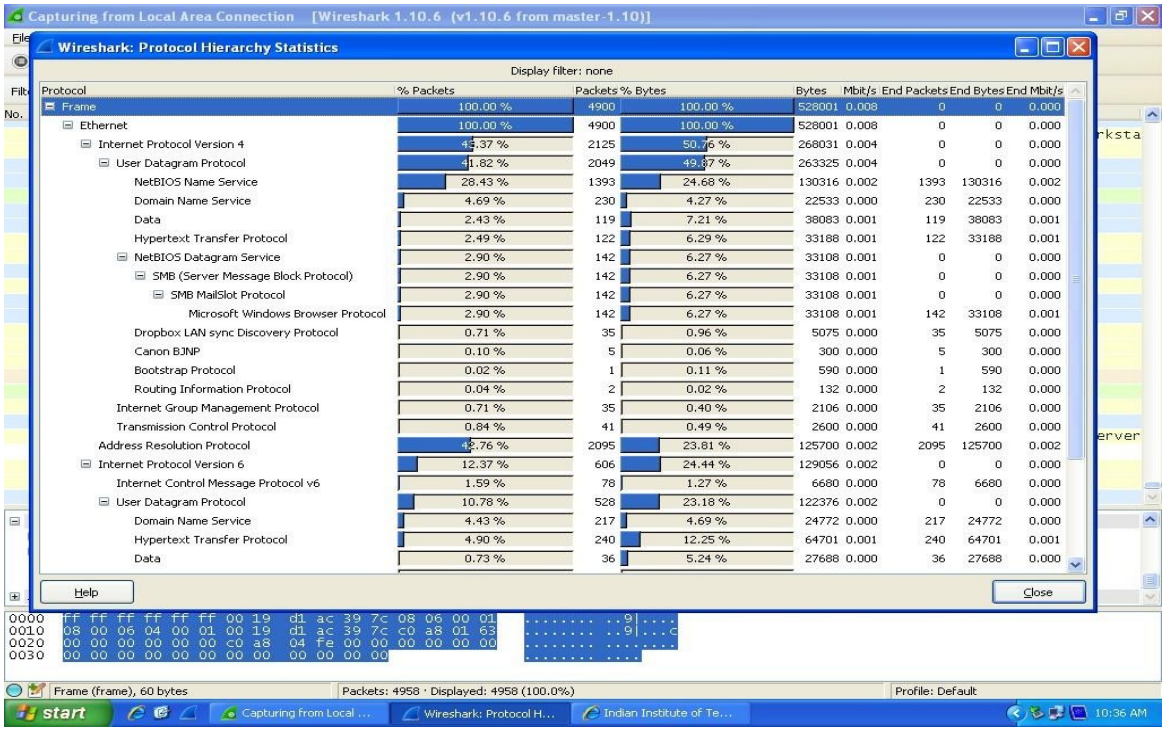
1: Main Window



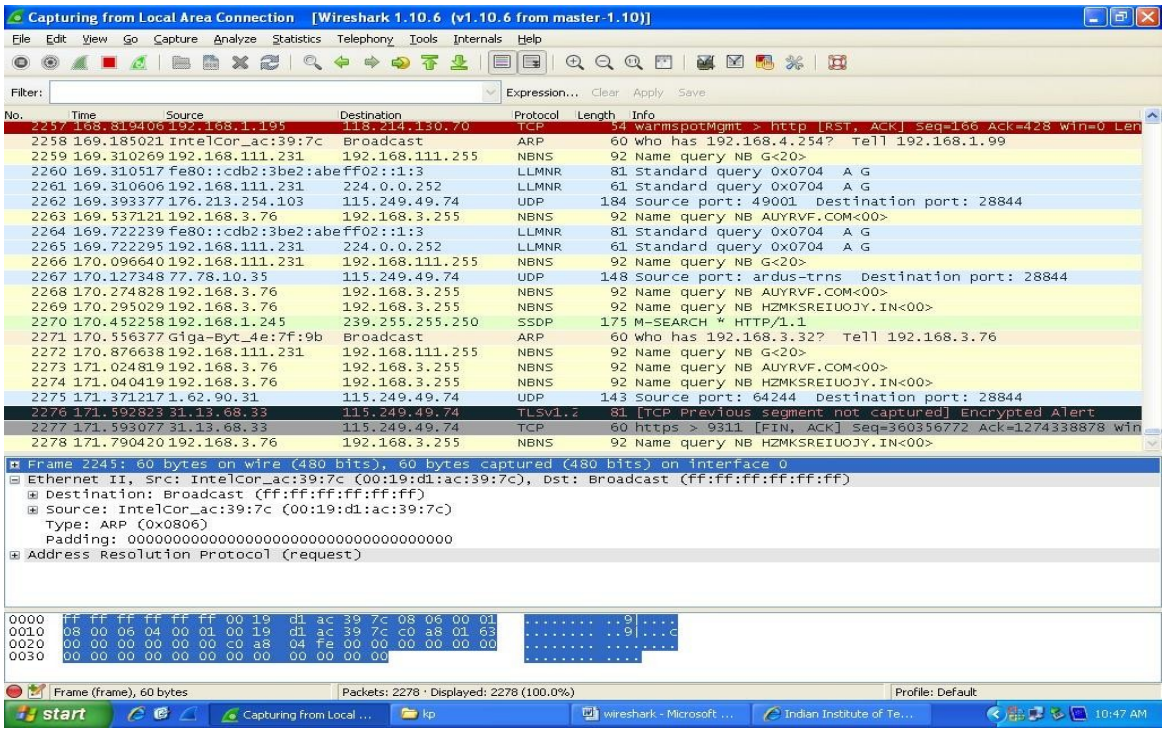
2. Capture Option Window



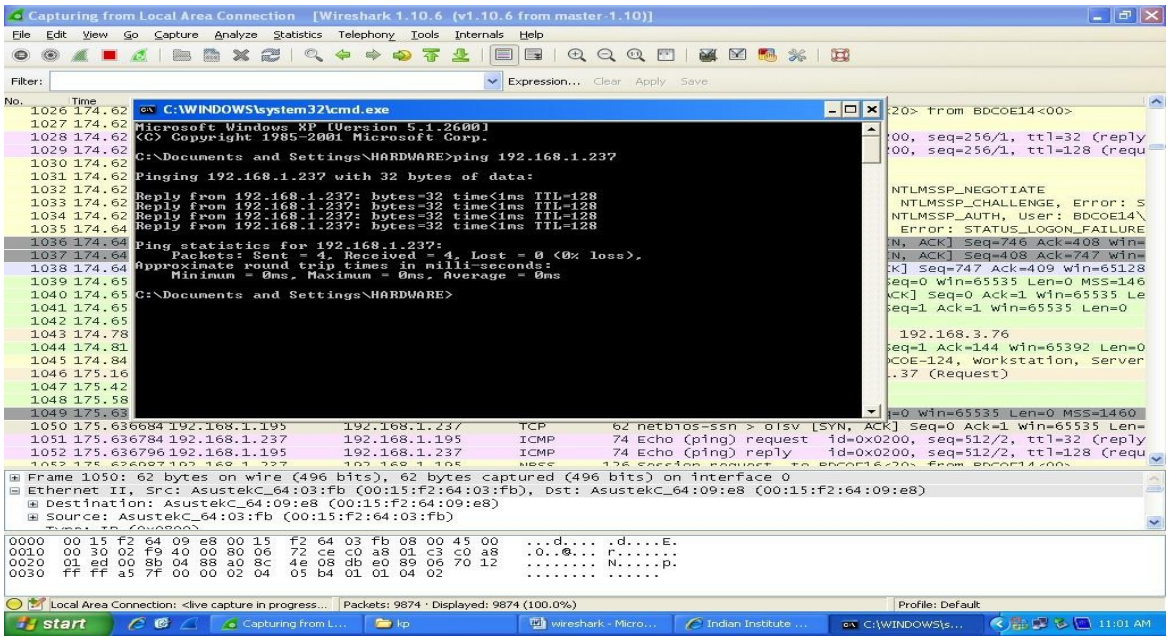
3. Protocol hierarchy



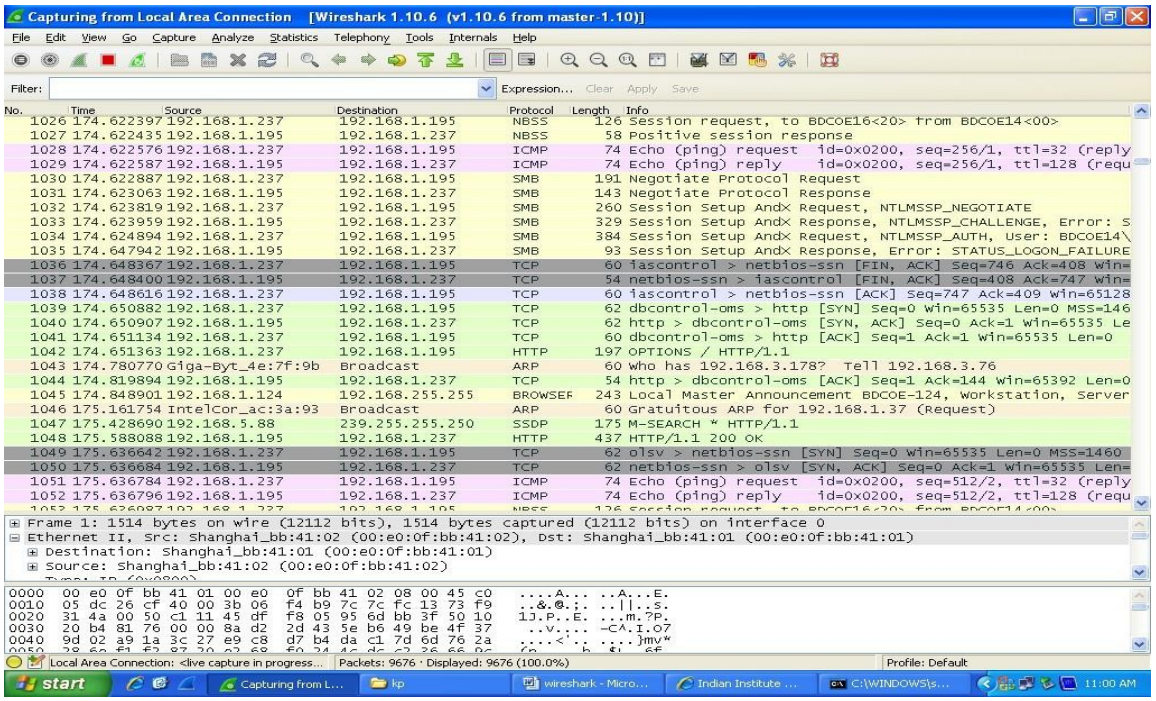
4.window showing packet list plane, packet detail plain, Packet byte plane



Task 1: Ping PDU capture Step1:



Step2:Window showing packet list plane for echo request and reply



From the Wireshark Packet List answer the following:

1. What protocol is used by ping? **ICMP**

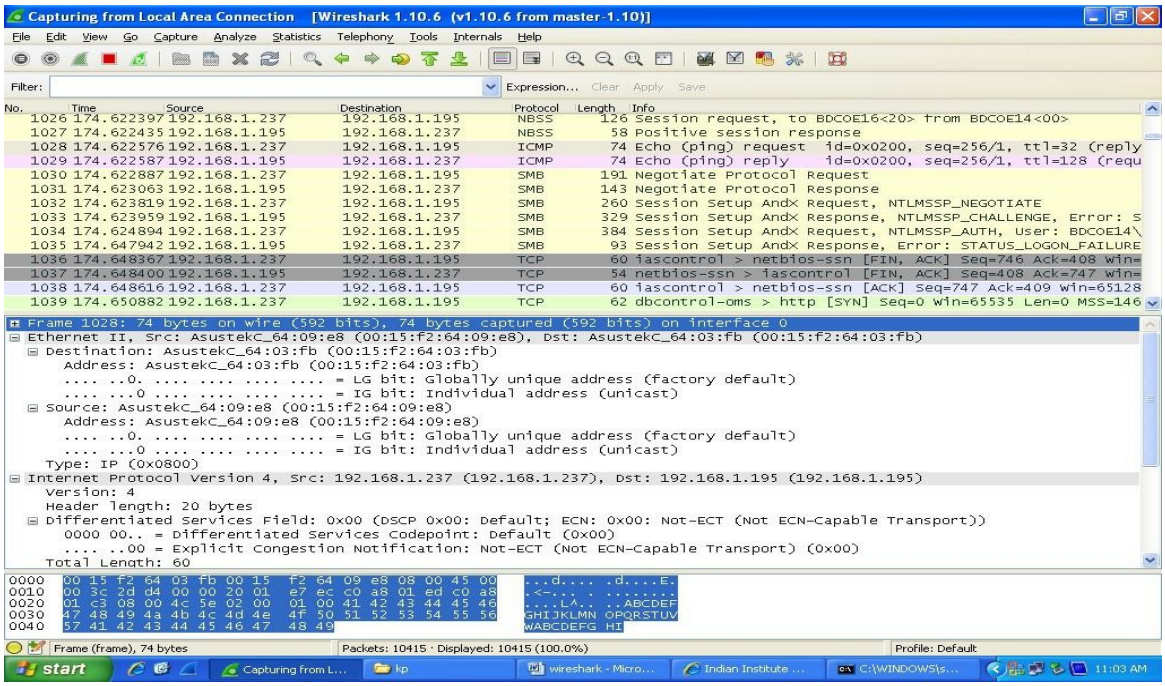
2. What is the full protocol name? **Internet Control Message Protocol**

3. What are the names of the two ping messages? **1.Echo (ping) request**

2.Echo (ping) reply

4. Are the listed source and destination IP addresses what you expected? Yes / No Why? **Yes. Because source address (request) is192.168.1.195 and destination address (reply) is 192.168.1.237**.

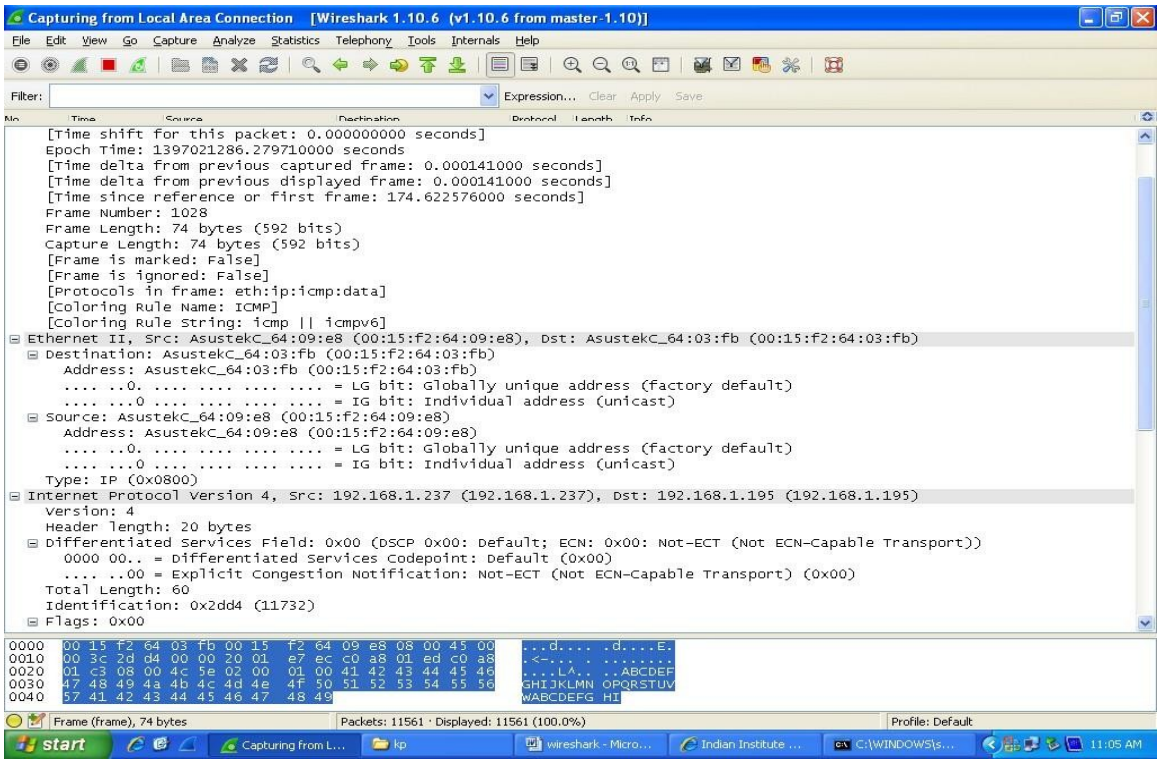
Step 3: packet detail plain for echo request



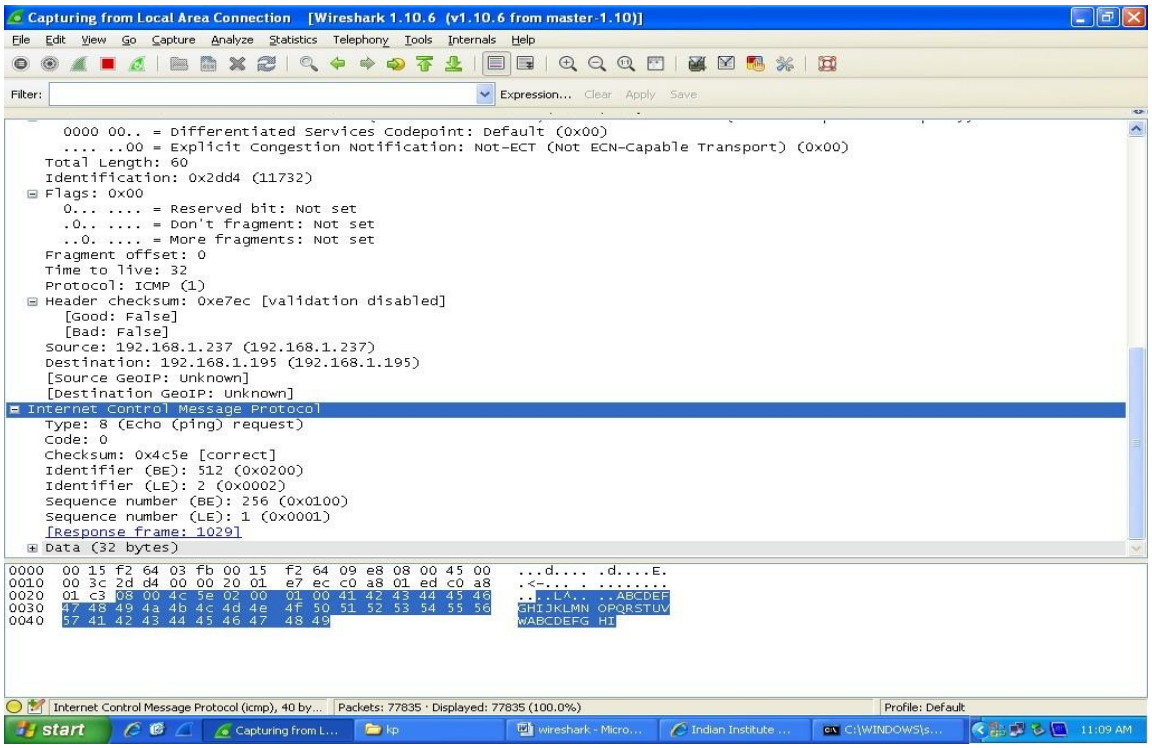
What protocols are in the Ethernet frame?

IP i.e. Internet Protocol

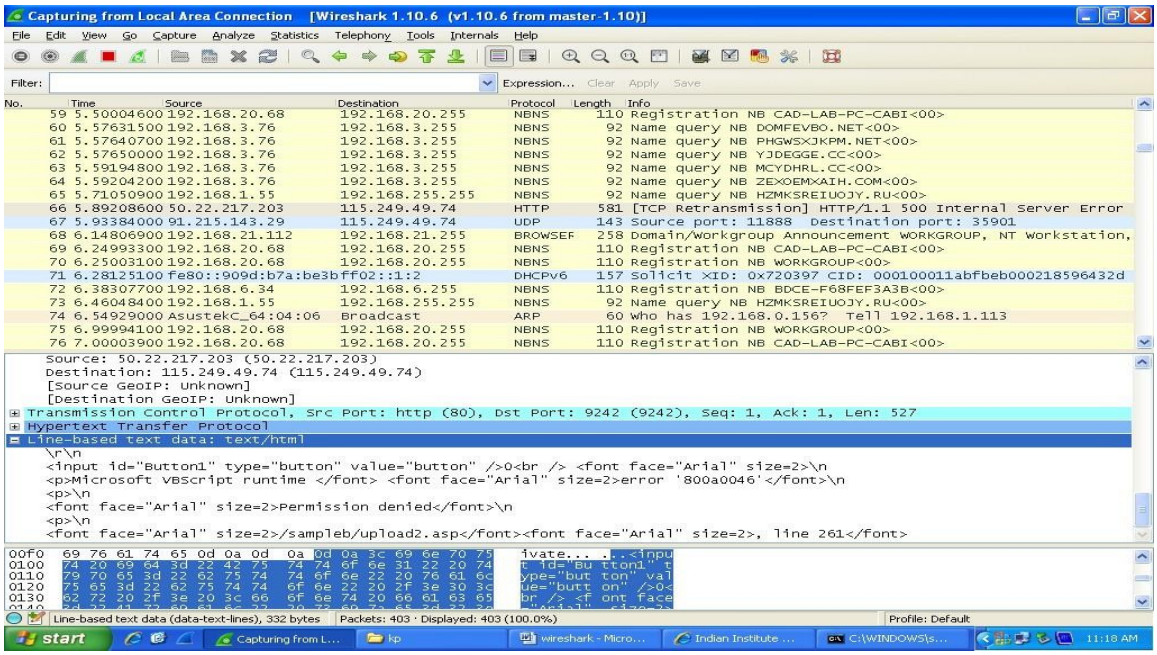
Window showing source and destination:



Window showing details in packet bytes plane:

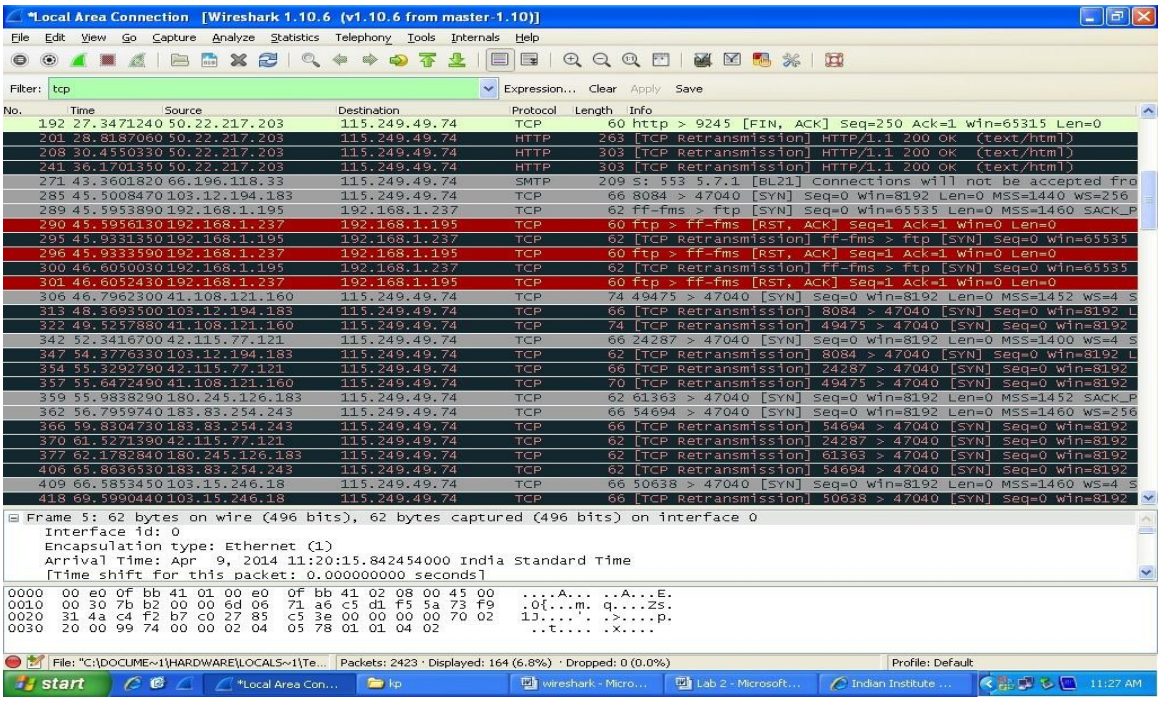


Window showing HTTP PDU capture:

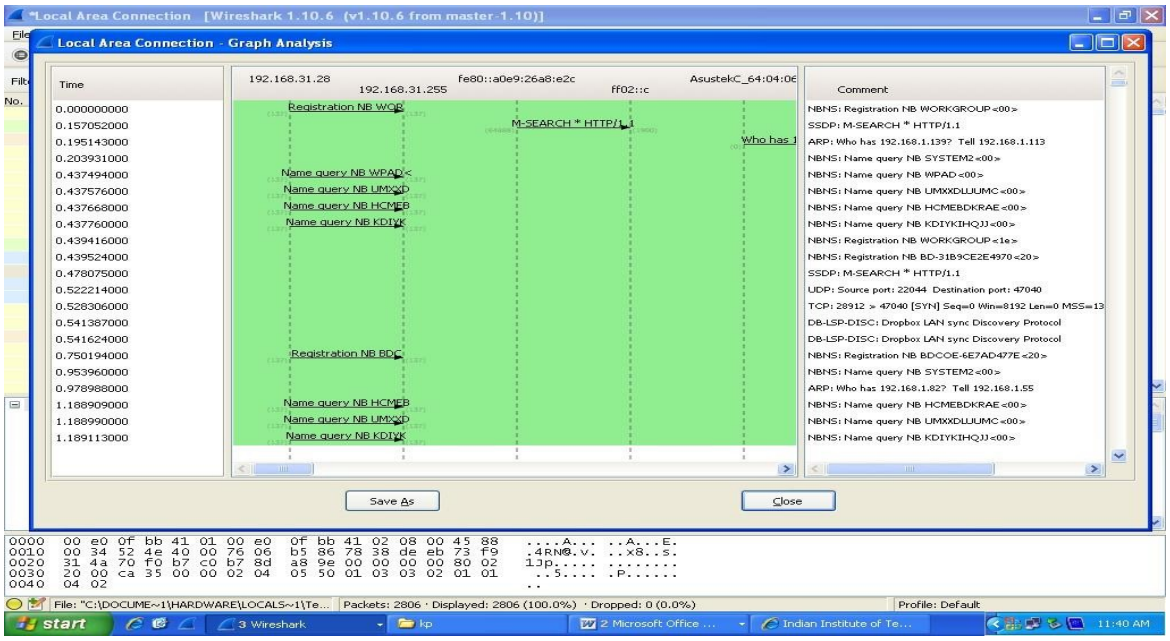


In the Packet Detail pane click on the "+" next to "**Line-based text data: html**"
When this information expands what is displayed? **HTML script is displayed**.

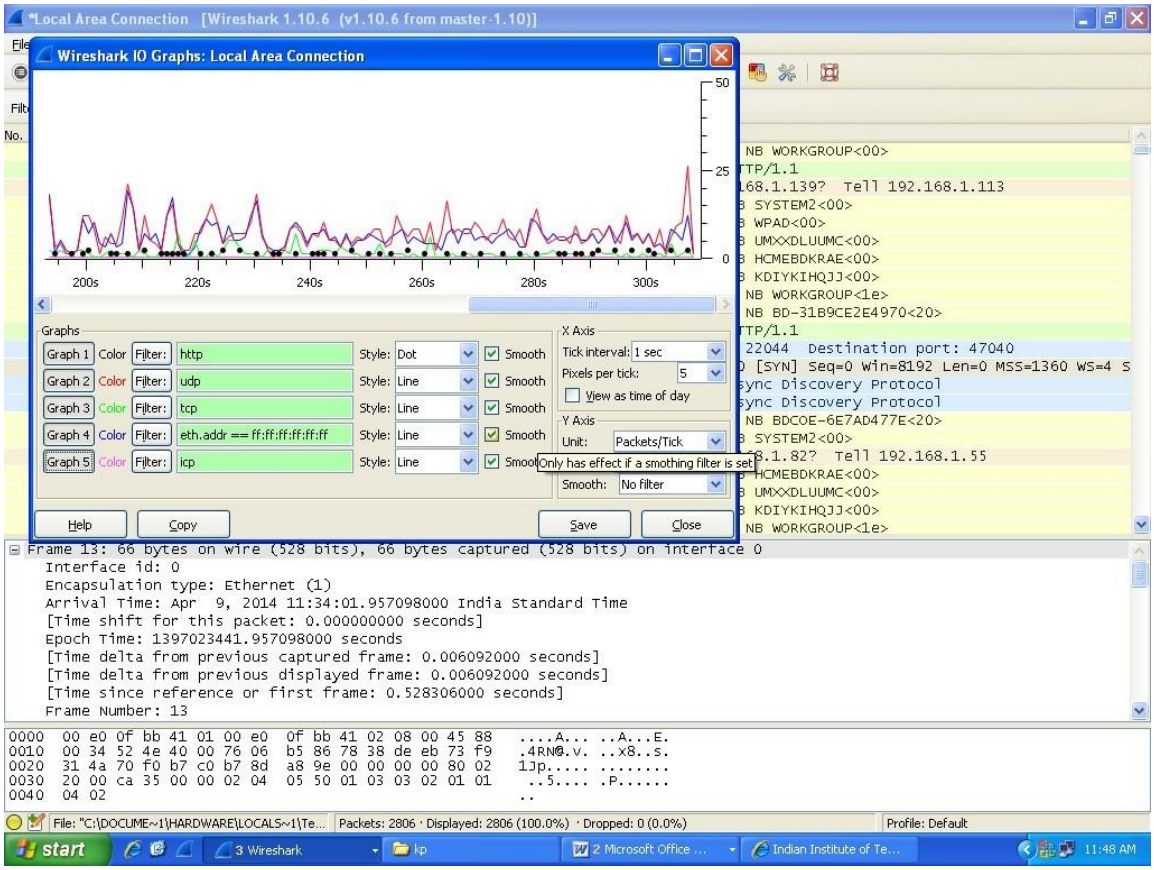
2.Window showing all TCP packets



Flow Graph:


Local Area Connection - Graph Analysis
Time
192.168.31.28
192.168.31.255
fe80::a0e9:26a8:e2c
ff02::c
AsustekC_64:04:0E
Comment
Registration NB WQB
M-SEARCH * HTTP/1.1
Name query NB WPAD<
Name query NB UMXXD
Name query NB HCMEB
Name query NB KDIYK
Registration NB BDC
Save As
Close

IO graph:


Wireshark IO Graphs: Local Area Connection
Graphs
Graph 1 Color Filter: http Style: Dot Smooth
Graph 2 Color Filter: udp Style: Line Smooth
Graph 3 Color Filter: tcp Style: Line Smooth
Graph 4 Color Filter: eth.addr == ff:ff:ff:ff:ff:ff Style: Line Smooth
Graph 5 Color Filter: icp Style: Line Smooth
X Axis
Tick interval: 1 sec
Pixels per tick: 5
View as time of day
Y Axis
Unit: Packets/Tick
Smooth: No filter
Only has effect if a smothing filter is set
Help
Copy
Save
Close
Frame 13: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
Interface id: 0
Encapsulation type: Ethernet (1)
Arrival Time: Apr 9, 2014 11:34:01.957098000 India Standard Time
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1397023441.957098000 seconds
[Time delta from previous captured frame: 0.006092000 seconds]
[Time delta from previous displayed frame: 0.006092000 seconds]
[Time since reference or first frame: 0.528306000 seconds]
Frame Number: 13